

FORECASTING THE PETROLEUM PRODUCTION USING MACHINE LEARNING

Team Members: Radha Rani Kommineni, Praveen Muppavarapu, Srilakshmi Mannam, Sai Sriram Kondrumutla, Naveen Kumar Bandela, Sai Kumar Reddy Gopireddy

ABSTRACT:

The fundamental for evaluating oil reservoir states and establishing oilfield growth plans is reliable crude oil output projection. The task of estimating future values of a particular sequence using previous data is known as time series forecasting (TSF). With the growing availability of large volumes of records and the necessity for detailed production forecasting, a powerful forecasting technique that infers the statistical dependency between past and future values is critical.

INTRODUCTION:

Machine learning is one of the applications of artificial intelligence (AI) that provides computers, the ability to learn automatically and improve from experience instead of being explicitly programmed. It focuses on developing computer programs that can access data and use it to learn from themselves. Forecasting petroleum production is a very pertinent task in the petroleum industry, where the accurate estimation of petroleum reserves involves a massive investment of money, time, and technology in the context of a wide range of operating and maintenance scenarios. performing accurate production forecasting, particularly a powerful forecasting technique infers the stochastic dependency between past and future values is highly needed. Time series forecasting (TSF) is the task of predicting future values of a given sequence using historical data from oil production. Recently, this task has attracted the attention of researchers in the area of machine learning to address the limitations of traditional forecasting methods, which are time-consuming and full of complexity.

LITERATURE REVIEW:

Machine Learning can be essentially characterized as a subset of Artificial intelligence as it enables devices to gain from their experiences and improve themselves without doing any coding. It is the investigation of making machines increasingly human-like in their conduct and choices by enabling them to learn and build up their own projects through least human mediation. The main aim of

machine learning is to allow computers to learn automatically without human intervention and adjust actions accordingly.

Reference Papers:

[http://iosrjen.org/Papers/vol3_issue12%20\(part-1\)/F031214455.pdf?msclkid=2a7a40fccff711ec84aeb7a8534337ec](http://iosrjen.org/Papers/vol3_issue12%20(part-1)/F031214455.pdf?msclkid=2a7a40fccff711ec84aeb7a8534337ec)

https://www.researchgate.net/publication/342452307_Time_Series_Analysis_of_Crude_Oil_Production_in_Nigeria

<https://www.jstor.org/stable/41322296?msclkid=2a7a7426cff711ec9b23ab176e06799d>

MACHINE LEARNING METHODS:

Machine learning algorithms are often categorized as supervised and unsupervised.

Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events. Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function to make predictions about the output values. The system is able to provide targets for any new input after sufficient training. The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

In contrast, unsupervised machine learning algorithms are used when the information used to train is neither classified nor labeled. Unsupervised learning studies how systems can infer a function to describe a hidden structure from unlabeled data. The system doesn't figure out the right output, but it explores the data and can draw inferences from datasets to describe hidden structures from unlabeled data.

Semi-supervised machine learning algorithms fall somewhere in between supervised and unsupervised learning since they use both labeled and unlabeled data for training – typically a small amount of labeled data and a large amount of unlabeled data. The systems that use this method are able to considerably improve learning accuracy. Usually, semi-supervised learning is chosen when the acquired labeled data requires skilled and relevant resources in order to train it / learn from it. Otherwise, acquiring unlabeled data generally doesn't require additional resources.

Reinforcement machine learning algorithms are a learning method that interacts with its environment by producing actions and discovering errors or rewards. Trial and error search and delayed reward are the most relevant characteristics of reinforcement learning. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Simple reward feedback is required for the agent to learn which action is best. This is known as the reinforcement signal.

APPLICATIONS OF MACHINE LEARNING:

1. Virtual Personal Assistants
2. Predictions while Commuting
3. Videos Surveillance
4. Social Media Services
5. Email Spam and Malware Filtering
6. Online Customer Support
7. Search Engine Result Refining
8. Product Recommendations
9. Online Fraud Detection

DATA:

DATASET BACKGROUND AND QUALITY:

Data Sources: <https://www.kaggle.com/>

No. of datasets: 2

Quantity: petroleumflowdata.csv(5308 rows*8 columns)

This dataset is used to predict total flow rate

L9									
	A	B	C	D	E	F	G	H	
1	Date	Cum. flow	Condensate	Water	CGR	WGR	Avg. Pressure	Total flow	
2	22/05/2004	10.7415	0	0	0	0	0	10.7415	
3	23/05/2004	34.3655	8.05	3.2	0.340755	0.135455	2242	23.624	
4	24/05/2004	58.606	8.12	5.03	0.334977	0.207504	2283	24.2405	
5	25/05/2004	83.7587	8.01	3.1	0.318455	0.123247	2323	25.1527	
6	26/05/2004	108.0027	5.58	1.43	0.23016	0.058984	2328.5	24.244	
7	27/05/2004	133.1326	5.62	1.78	0.223638	0.070832	2320	25.1299	
8	28/05/2004	158.0935	5.6	1.35	0.224351	0.054085	2299	24.9609	
9	29/05/2004	181.2394	5.35	1.1	0.231142	0.047525	2304	23.1459	
10	30/05/2004	200.0078	4.95	1.2	0.263741	0.063937	2317.5	18.7684	
11	31/05/2004	220.0397	4.9	1.4	0.24461	0.069889	2302.5	20.0319	
12	1/6/2004	239.6143	4.89	2.44	0.249814	0.124651	2267.5	19.5746	
13	2/6/2004	259.9414	5.15	2.65	0.253356	0.130368	2217.5	20.3271	
14	3/6/2004	281.3769	5.65	1.2	0.263581	0.055982	2222.5	21.4355	
15	4/6/2004	302.2643	5.45	1.2	0.260923	0.057451	2217.5	20.8874	
16	5/6/2004	322.7142	5.6	1.1	0.27384	0.05379	2225	20.4499	
17	6/6/2004	343.5337	6.1	1.5	0.292995	0.072048	2205	20.8195	
18	7/6/2004	365.438	9.9	1.6	0.451966	0.073045	2182.5	21.9043	
19	8/6/2004	386.8038	7.9	1.5	0.36975	0.070206	2188.5	21.3658	
20	9/6/2004	408.0354	9.1	1.5	0.428606	0.070649	2192.5	21.2316	
21	10/6/2004	429.3083	9.1	1.5	0.427774	0.070512	2194	21.2729	
22	11/6/2004	450.3749	8.06	1.5	0.382596	0.071203	2196.5	21.0666	
23	12/6/2004	470.8162	8.05	2	0.393811	0.097841	2190	20.4413	
24	13/06/2004	491.2075	8.45	1.5	0.414392	0.073561	2193.5	20.3913	
25	14/06/2004	511.4181	8.5	1.2	0.420571	0.059375	2191.5	20.2106	
26	15/06/2004	531.586	8.5	2.2	0.421462	0.109084	2196.5	20.1679	
27	16/06/2004	552.0619	8.75	2	0.427332	0.097676	2192	20.4759	
28	17/06/2004	572.2524	8.5	1.1	0.453554	0.053493	2205	20.8815	

gasproduction.csv(5338 rows*2 columns)

This dataset is used to predict oil production

	A	B
1	Time	Production
2	5/22/2004	10.7415
3	5/23/2004	23.624
4	5/24/2004	24.2405
5	5/25/2004	25.1527
6	5/26/2004	24.244
7	5/27/2004	25.1299
8	5/28/2004	24.9609
9	5/29/2004	23.1459
10	5/30/2004	18.7684
11	5/31/2004	20.0319
12	6/1/2004	19.5746
13	6/2/2004	20.3271
14	6/3/2004	21.4355
15	6/4/2004	20.8874
16	6/5/2004	20.4499
17	6/6/2004	20.8195
18	6/7/2004	21.9043
19	6/8/2004	21.3658
20	6/9/2004	21.2316
21	6/10/2004	21.2729
22	6/11/2004	21.0666
23	6/12/2004	20.4413
24	6/13/2004	20.3913
25	6/14/2004	20.2106
26	6/15/2004	20.1679
27	6/16/2004	20.4759
28	6/17/2004	20.8815

DATA PROCESSING, WRANGLING, AND EDA:

DATA PROCESSING:

data processing, manipulation of data by a computer. It includes the conversion of raw data to machine-readable form, the flow of data through the CPU and memory to output devices, and formatting or transformation of output. Any use of computers to perform defined operations on data can be included under data processing.

DATA PRE-PROCESSING:

Pre-processing refers to the transformations applied to our data before feeding it to the algorithm. Data Pre-processing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.

WRANGLING:

Data Wrangling is a technique that is executed at the time of making an interactive model. In other words, it is used to convert the raw data into a format that is convenient for the consumption of data. This technique is also known as Data Munging. This method also follows certain steps such as extracting the data from different data sources, sorting data using the certain algorithms performed, decomposing the data into a different structured format, and finally storing the data in another database.

EXPLORATORY DATA ANALYSIS (EDA):

Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods.

It helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions.

Technically, The primary motive of EDA is to

- Examine the data distribution
- Handling missing values of the dataset
- Handling the outliers
- Removing duplicate data

Importing all the python libraries that are required:

1. **NumPy** for numerical calculations and scientific computing
2. **Pandas** for handling data
3. **Matplotlib** and **Seaborn** for visualization.

```
In [1]: #importing required libraries
import pandas as pd          # DataFrames for tabular data
import numpy as np          # ndarrays for gridded data
import matplotlib.pyplot as plt # plotting
import seaborn as sns       # generates enhanced plots
```

Loading the data into the **Pandas** data frame

```
In [2]: #reading a .csv file as a DataFrame (petroleumflowdat.csv)
oil = pd.read_csv('petroleumflowdata.csv')
```

Observing the dataset by checking a few of the rows using the **head()** method, which returns the first five records from the dataset.

```
In [3]: # we using this command for a table preview
oil.head()
```

```
Out[3]:
```

	Date	Cum. flow	Condensate	Water	CGR	WGR	Avg. Pressure	Total flow
0	22/05/2004	10.7415	0.00	0.00	0.000000	0.000000	0.0	10.7415
1	23/05/2004	34.3655	8.05	3.20	0.340755	0.135455	2242.0	23.6240
2	24/05/2004	58.6060	8.12	5.03	0.334977	0.207504	2283.0	24.2405
3	25/05/2004	83.7587	8.01	3.10	0.318455	0.123247	2323.0	25.1527
4	26/05/2004	108.0027	5.58	1.43	0.230160	0.058984	2328.5	24.2440

Using **shape**, we can observe the dimensions of the data.

```
In [5]: oil.shape
```

```
Out[5]: (5307, 8)
```

info() method shows some of the characteristics of the data such as Column Name, No. of non-null values of our columns, Datatype of the data, and Memory Usage.

```
In [11]: oil.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5307 entries, 0 to 5306
Data columns (total 8 columns):
#   Column          Non-Null Count  Dtype
---  -
0   Date            5306 non-null   object
1   Cum. flow       5307 non-null   float64
2   Condensate      5307 non-null   float64
3   Water           5307 non-null   float64
4   CGR             5307 non-null   float64
5   WGR             5307 non-null   float64
6   Avg. Pressure   5307 non-null   float64
7   Total flow      5307 non-null   float64
dtypes: float64(7), object(1)
memory usage: 331.8+ KB
```

describe() shows summary statistical characteristics of each numerical feature (int64 and float64 types): number of non-missing values, mean, standard deviation, range, median, 0.25, 0.50, 0.75 quartiles.

```
In [7]: #check the summary statistics.
oil.describe()
```


```
Out[7]:
```

	Cum. flow	Condensate	Water	CGR	WGR	Avg. Pressure	Total flow
count	5307.000000	5307.000000	5307.000000	5307.000000	5307.000000	5307.000000	5307.000000
mean	83937.163711	21.602225	50.172154	0.718086	2.701395	1906.785598	29.811795
std	47152.673681	18.560487	76.061487	0.588929	5.286858	222.967627	10.352953
min	10.741500	0.000000	0.000000	0.000000	0.000000	0.000000	4.506700
25%	47144.646100	9.500000	4.750000	0.385384	0.113129	1731.500000	21.595100
50%	79235.443400	18.000000	6.000000	0.570292	0.165248	1850.000000	31.034400
75%	127310.681700	26.000000	76.250000	0.744793	3.755925	2066.750000	38.229250
max	158211.196400	110.500000	495.000000	4.652305	40.082697	2595.000000	51.660900

Checking for missing values

```
In [8]: # check missing values  
oil.isnull().sum()
```

```
Out[8]: Date          1  
Cum. flow          0  
Condensate         0  
Water              0  
CGR                0  
WGR                0  
Avg. Pressure      0  
Total flow         0  
dtype: int64
```



The 'Date' column has 1 missing value, By checking the missing date we can see it corresponds to the last day in our time series record since it is just one data point and given it is the last in our record(hence does not have predictive value) we just drop it.

```
In [9]: oil = oil.dropna(axis = 0)
```

Checking whether the missing values are handled or not

```
In [14]: # check for missing values again after dropping them  
oil.isnull().sum()
```

```
Out[14]: Date          0  
Cum. flow          0  
Condensate         0  
Water              0  
CGR                0  
WGR                0  
Avg. Pressure      0  
Total flow         0  
dtype: int64
```

We can check for duplicate values in our dataset as the presence of duplicate values will hamper the accuracy of our ML model.


```
In [16]: #Checking for the duplicated values
duplicatedvalues=oil.duplicated()
print(duplicatedvalues.sum())
oil[duplicatedvalues]
```

0

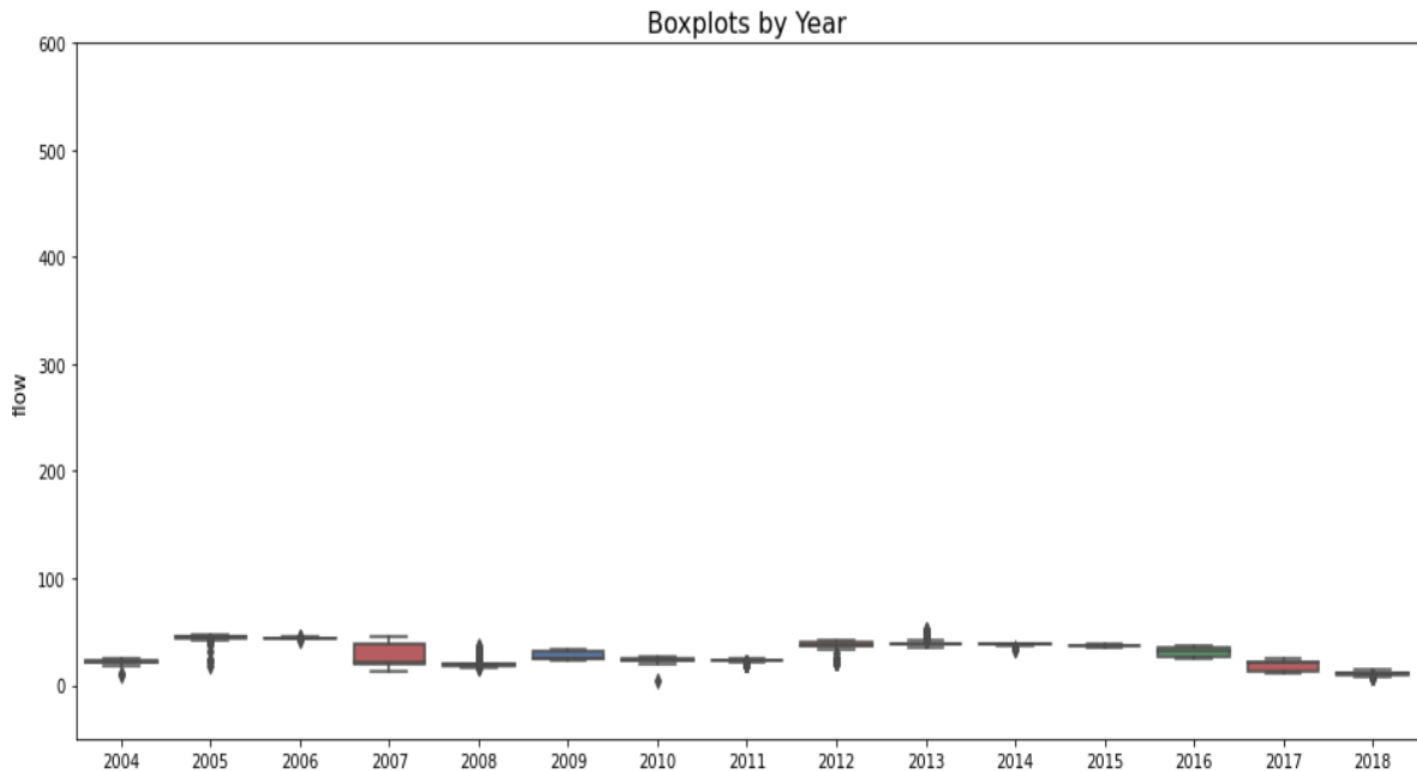
Out[16]:

Date	Cum. flow	Condensate	Water	CGR	WGR	Avg. Pressure	Total flow
------	-----------	------------	-------	-----	-----	---------------	------------

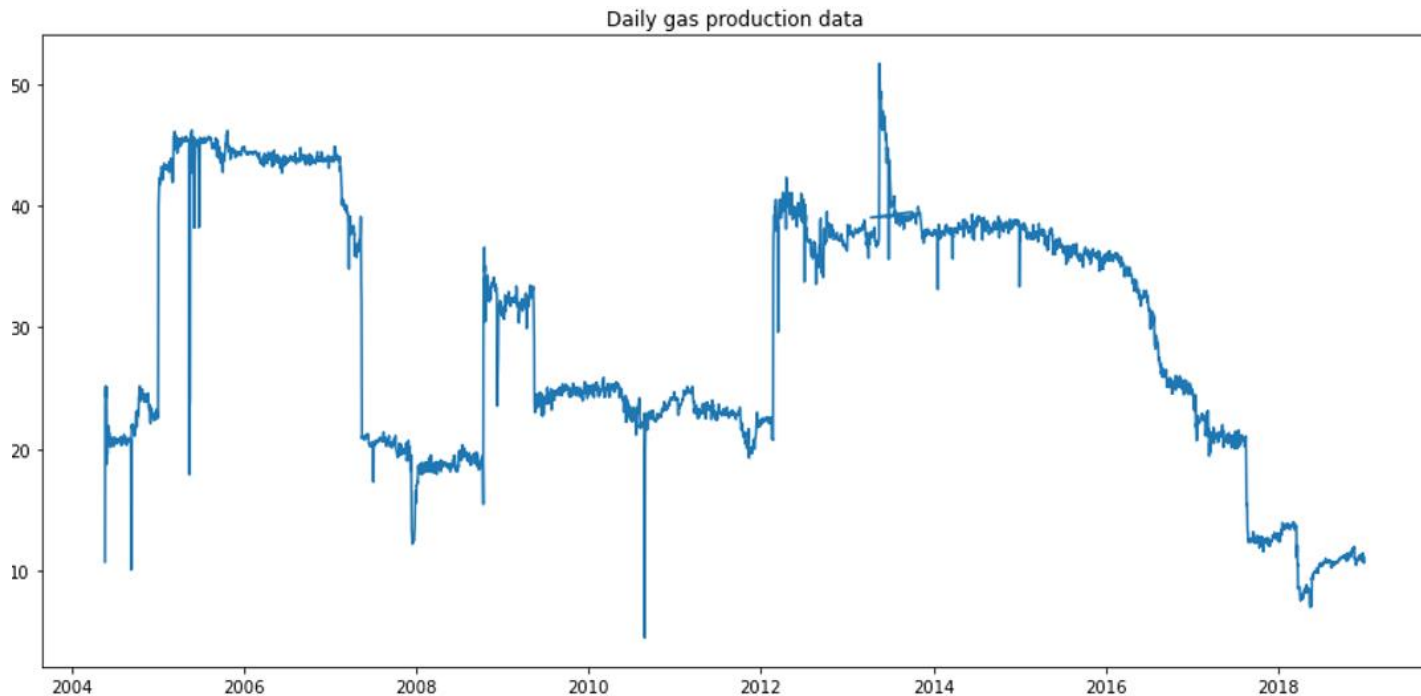
NO DUPLICATES - So, no need for handling them

Similarly, performing the same steps for the other dataset(gasproduction.csv)

Box plots showing the rate of oil flow for every year.



Graph showing the fluctuations in gas productions with time.



METHODOLOGY:

METHODS DESCRIPTION:

Machine Learning Methods and Algorithms

1. XG boosting
2. Random Forest
3. Lasso regression
4. Recurrent Neural Network (RNN)

1. XG BOOSTING:

XGBoost, which stands for Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree (GBDT) machine learning library. It provides a parallel tree boosting and is the leading machine learning library for regression, classification, and ranking problems.

It's vital to understand XGBoost to first grasp the machine learning concepts and algorithms that XGBoost builds upon supervised machine learning, decision trees, ensemble learning, and gradient boosting.

Supervised machine learning uses algorithms to train a model to find patterns in a dataset with labels and features and then uses the trained model to predict the labels on a new dataset's features.

2. RANDOM FOREST:

Random Forests are an ensemble learning method (also thought of as a form of nearest neighbor predictor) for classification and regression techniques. It builds multiple decision trees and then merges them together in order to get more accurate and stable predictions. It constructs a number of Decision trees at training time and outputs the class that is the mode of the classes output by individual trees. It also tries to minimize the problems of high variance and high bias by averaging to find a natural balance between the two extremes. Both R and Python have robust packages to implement this algorithm.

3. LASSO REGRESSION:

Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point as they mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters). This particular type of regression is well-suited for models showing high levels of multicollinearity or when you want to automate certain parts of model selection, like variable selection/parameter elimination.

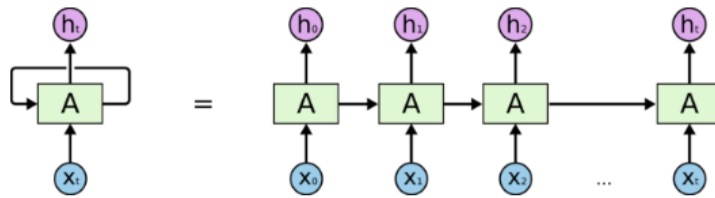
Performing the Regression

Lasso solutions are quadratic programming problems, which are best solved with software (like Matlab). The goal of the algorithm is to minimize:

$$\sum_{i=1}^n (y_i - \sum_j x_{ij} \beta_j)^2 + \lambda \sum_{j=1}^p |\beta_j|$$

4. RNN/LSTM:

A recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time-series data and where connections between nodes form a directed or undirected graph along a temporal sequence. It is a generalization of a feedforward neural network that has an internal memory That allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable-length sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition, or speech recognition. RNNs can use their internal state (memory) to process sequences of inputs



An unrolled recurrent neural network.

ADVANTAGES OF RECURRENT NEURAL NETWORK:

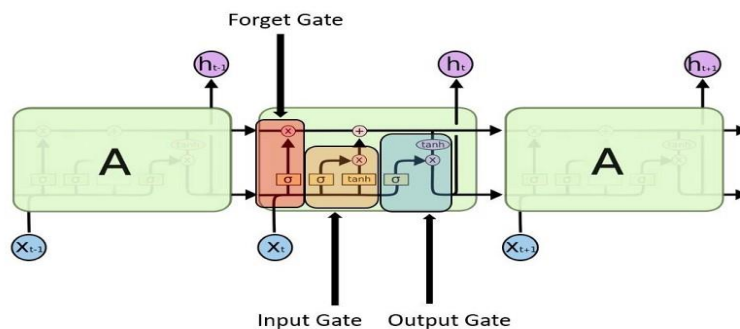
1. RNN can model sequence of data so that each sample can be assumed to be dependent on previous ones
2. Recurrent neural network is even used with convolutional layers to extend the effective pixel neighborhood.

DISADVANTAGES OF RECURRENT NEURAL NETWORK:

1. Gradient vanishing and exploding problems.
2. Training an RNN is a very difficult task.
3. It cannot process very long sequences if using tanh or relu as an activation function.

LONG SHORT-TERM MEMORY (LSTM):

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back-propagation. In an LSTM network, three gates are present.



1.**Input gate** — discover which value from input should be used to modify the memory. Sigmoid function decides which values to let through 0,1. and tanh function gives weightage to the values which are passed deciding their level of importance ranging from-1 to 1.

$$i_t = \sigma (W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

2.**Forget gate** — discover what details to be discarded from the block. It is decided by the sigmoid function. it looks at the previous state(ht-1) and the content input (Xt) and outputs a number between 0(omit this) and 1(keep this) for each number in the cell state Ct-1.

$$f_t = \sigma (W_f \cdot [h_{t-1}, x_t] + b_f)$$

4.**Output gate** — the input and the memory of the block is used to decide the output. Sigmoid function decides which values to let through 0,1. and tanh function gives weightage to the values which are passed deciding their level of importance ranging from-1 to 1 and multiplied with the output of Sigmoid.

$$o_t = \sigma (W_o [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh (C_t)$$

RESULTS AND DISCUSSION

MODEL EVALUATIONS AND METRICS:

Errors are usually measured in order to estimate the forecasting precision and performance evaluation of the forecasts. Here we are calculating RMSE, MSE, R Squared error and MAE.

- Root Mean Squared Error:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i^{obs} - y_i^{pred})^2},$$

Where, y_i obs is the current observation and \hat{y}_i is its predicted value.

- Mean Squared Error:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y})^2$$

- Mean Absolute Error:

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}|$$

- R squared(R^2):

The coefficient of determination or R-squared represents the proportion of the variance in the dependent variable which is explained by the linear regression model. It is a scale-free score i.e., irrespective of the values being small or large, the value of R square will be less than one.

$$R^2 = 1 - \frac{\sum (y_i - \hat{y})^2}{\sum (y_i - \bar{y})^2}$$

Where,

\hat{y} – predicted value of y

\bar{y} – mean value of y

The lower value of MAE, MSE, and RMSE implies higher accuracy of a regression model. However, a higher value of R square is considered desirable.

CORRELATION MATRIX:

A correlation matrix is a table showing correlation coefficients between variables. Each cell in the table shows the correlation between two variables.

A correlation matrix is used to summarize data, as an input into a more advanced analysis, and as a diagnostic for advanced analyses.

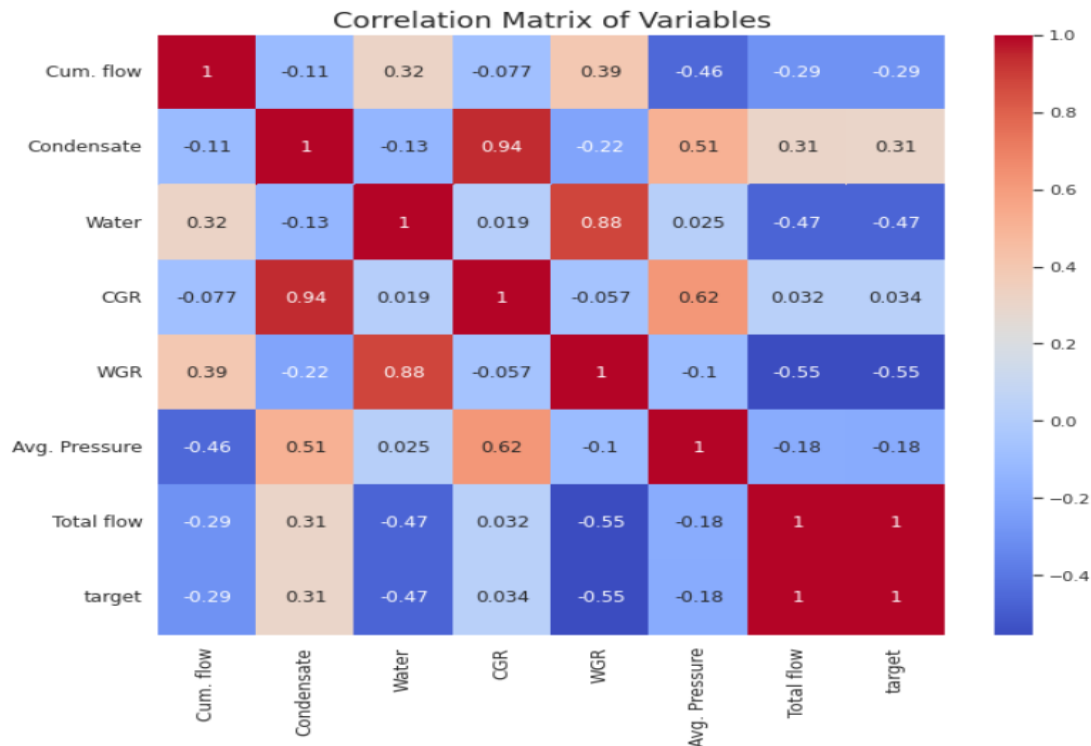


Fig: Correlation Matrix

There are generally weak correlations between the continuous explanatory variables with the "target" or "Total flow" variable one-period ahead. Also there are high correlations between "water" and "WGR", "condensate" and "CGR", "average pressure" and "CGR". This will not typically be a problem for machine learning models, but still would prefer a more parsimonious model.

Consequently, deleting the "CGR" and "WGR" variables.

```
In [29]: # Dropping unnecessary and redundant variables...
train_df = train_df.drop(['CGR', 'WGR'], axis=1)
```

Viewing the data after dropping the columns

```
In [30]: train_df.tail()
```

Out[30]:

	Date	Cum. flow	Condensate	Water	Avg. Pressure	Total flow	Day	Month	Year	target
5300	2018-11-25	158146.8371	3.75	310.0	1556.0	10.5183	25	11	2018	10.6977
5301	2018-11-26	158157.5348	3.75	338.0	1556.0	10.6977	26	11	2018	10.8203
5302	2018-11-27	158168.3551	4.75	312.0	1560.5	10.8203	27	11	2018	10.4677
5303	2018-11-28	158178.8228	5.75	265.0	1590.0	10.4677	28	11	2018	10.8618
5304	2018-11-29	158189.6846	3.50	293.0	1592.5	10.8618	29	11	2018	10.7708

OUTPUTS:

RANDOM FOREST REGRESSION:

The mean squared error is: 48.04671653212468

The R² is: 4.70441784208596

The mean absolute error is: 0.08565199042547356

The root_mean square error is: 6.93157388564276

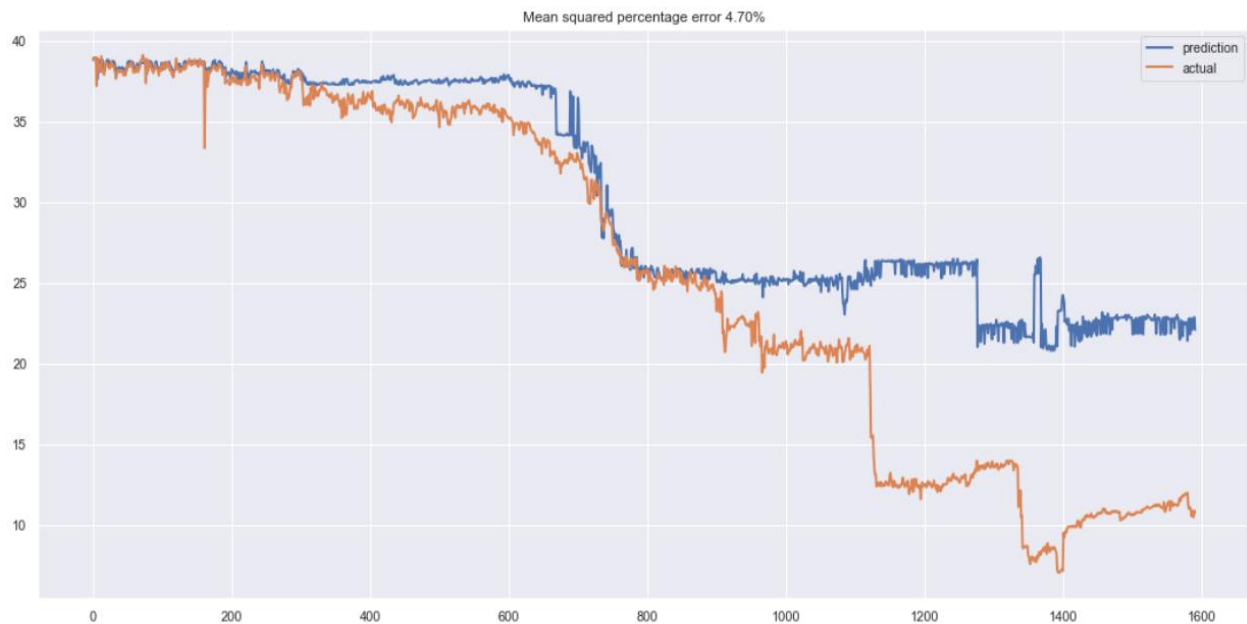


Fig: Actual and Predicted values/data using Random Forest Regression

XG BOOSTING:

The mean squared error is: 4.235288644007535

The R² is: 17.937669898059188

The mean absolute error is: 0.7298125944371332

The root_mean square error is: 2.5567602684864448

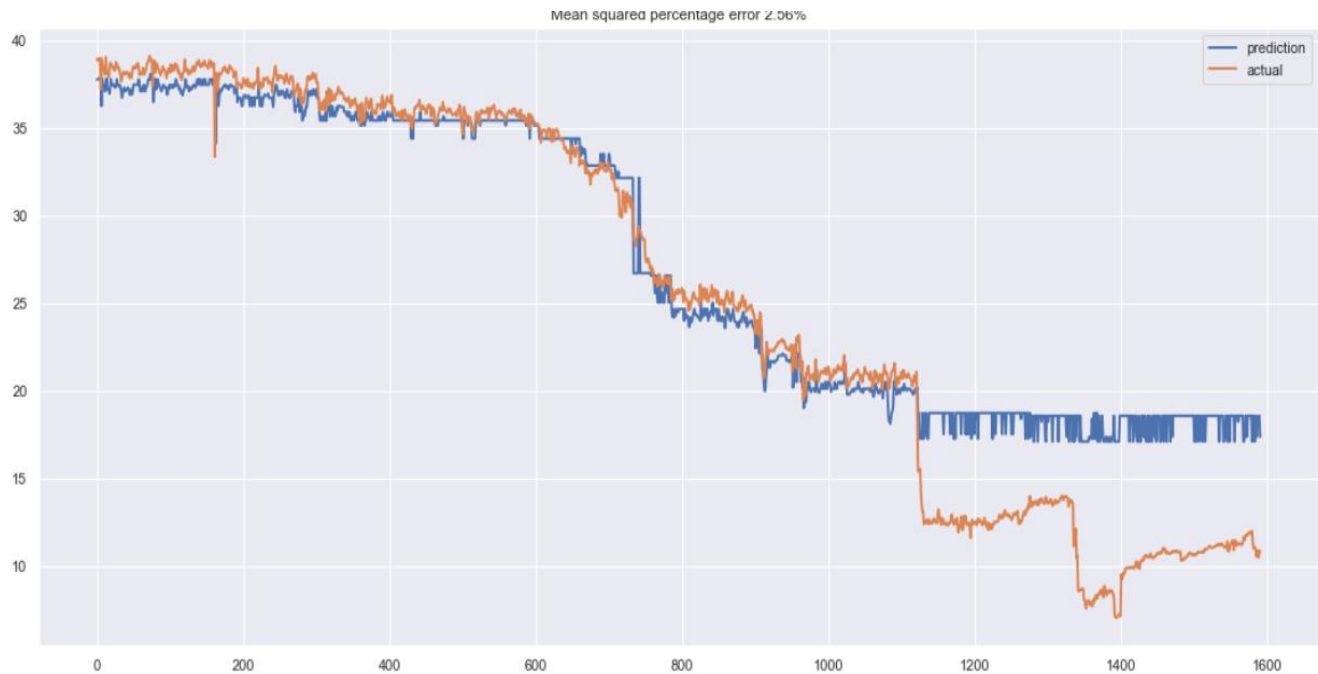


Fig: Actual and Predicted values/data using Extreme Gradient Boosting

LASSO REGRESSION:

The mean squared error is: 0.15070679172991105

The R2 is: 0.22844409944762595

The mean absolute error is: 0.9987073047681446

The root_mean square error is: 0.38820972647515034

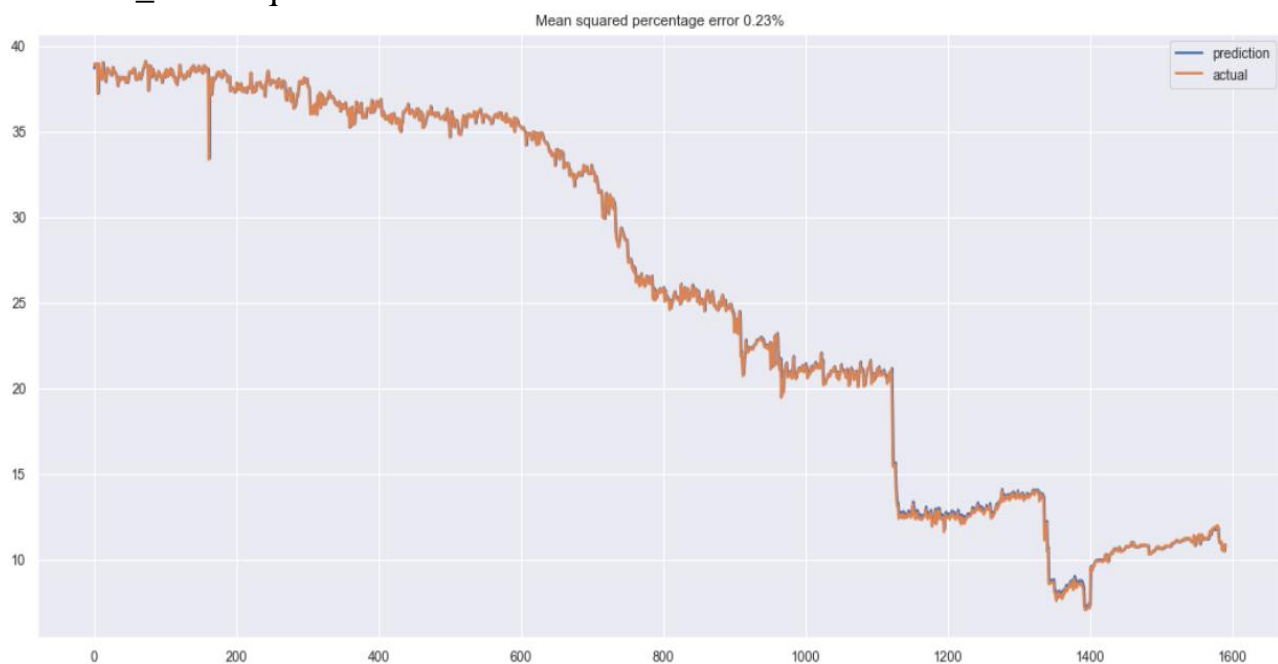


Fig: Actual and Predicted values/data using Lasso Regression

RNN/LSTM:

The mean squared error is: 0.57070047

The R² is: 0.5146727

The mean absolute error is: 0.9949156827558292

The root_mean square error is: 0.7554472

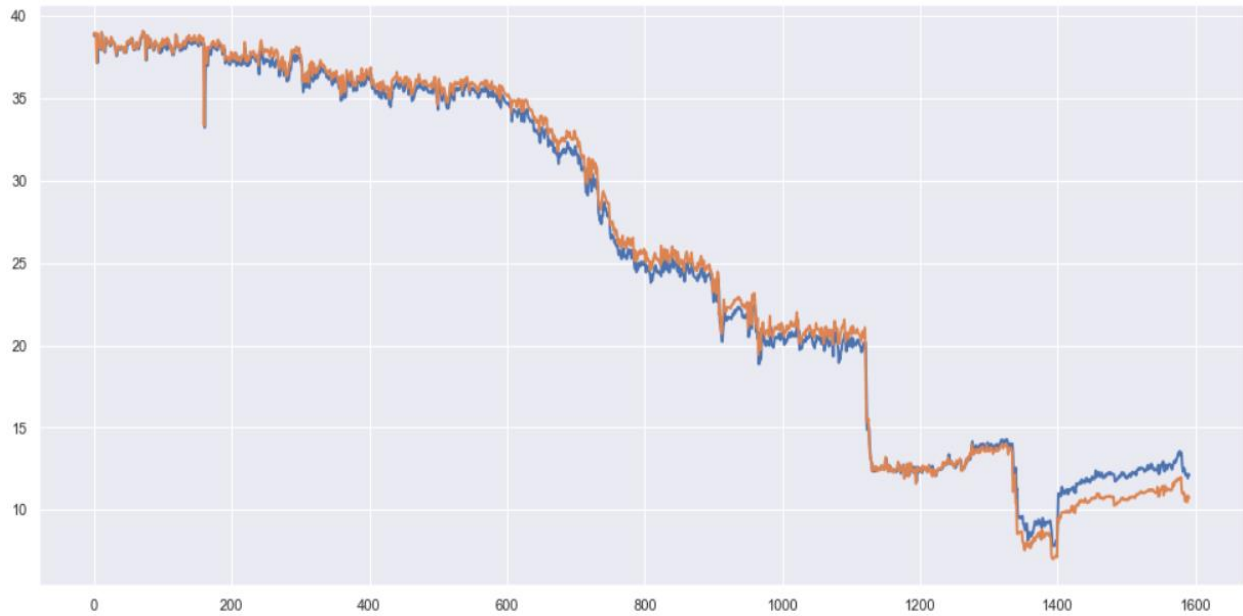


Fig: Actual and Predicted values/data using Recurrent Neural Network

COMPARISION OF MODEL EVALUATION METRICS FOR TOTAL FLOW RATE:

Models	RMSE	MSE	R ²	MAE
RNN-LSTM	0.755447	0.5707	0.514672	0.994915
Lasso	0.388209	0.150706	0.228444	0.998707
Random Forest	6.931573	48.046716	4.704417	0.085651
XGBoost	2.55676	4.235288	17.937669	0.729812

RESULTS FOR GAS PRODUCTION

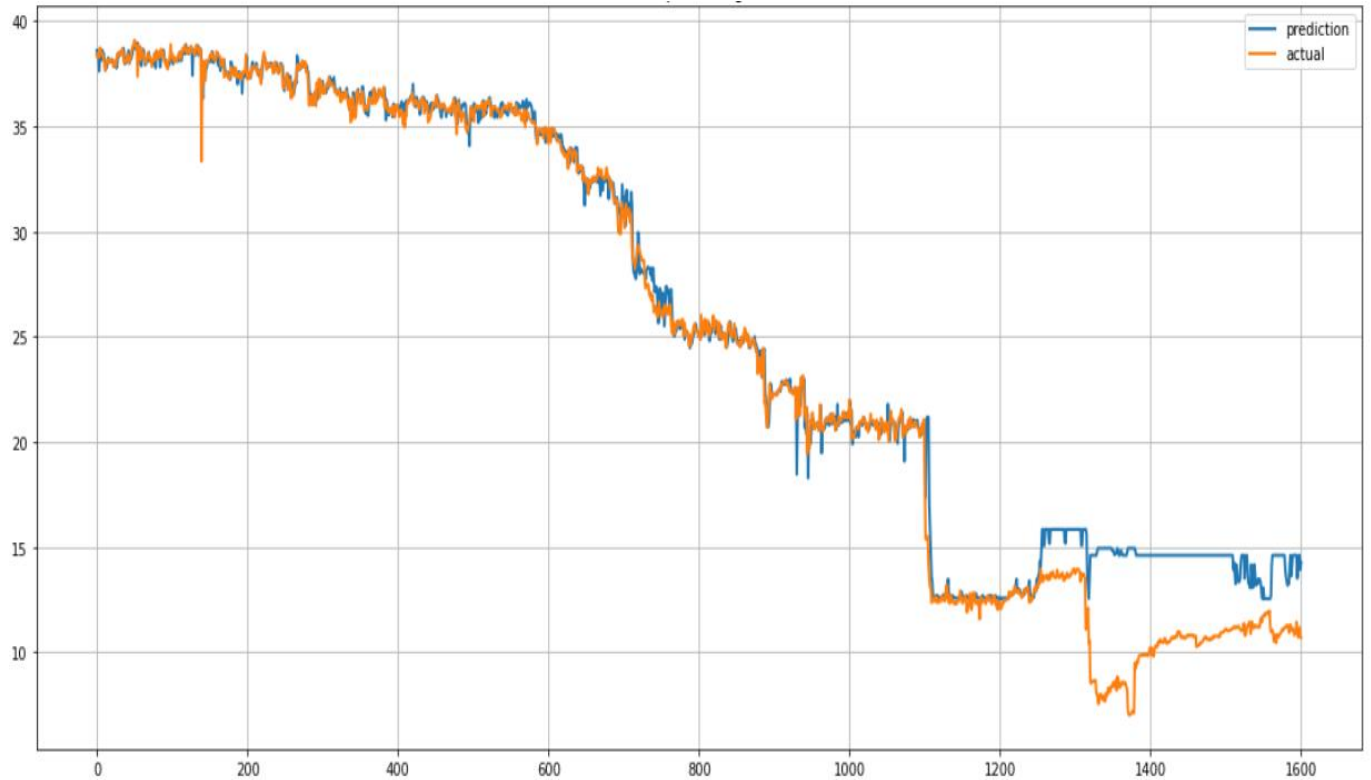
RANDOM FOREST:

The mean squared error is: 3.989972462671579

The R² is: 1.0860851022923417

The mean absolute error is: 0.9586147394976056

The root_mean square error is: 1.9974915425782356



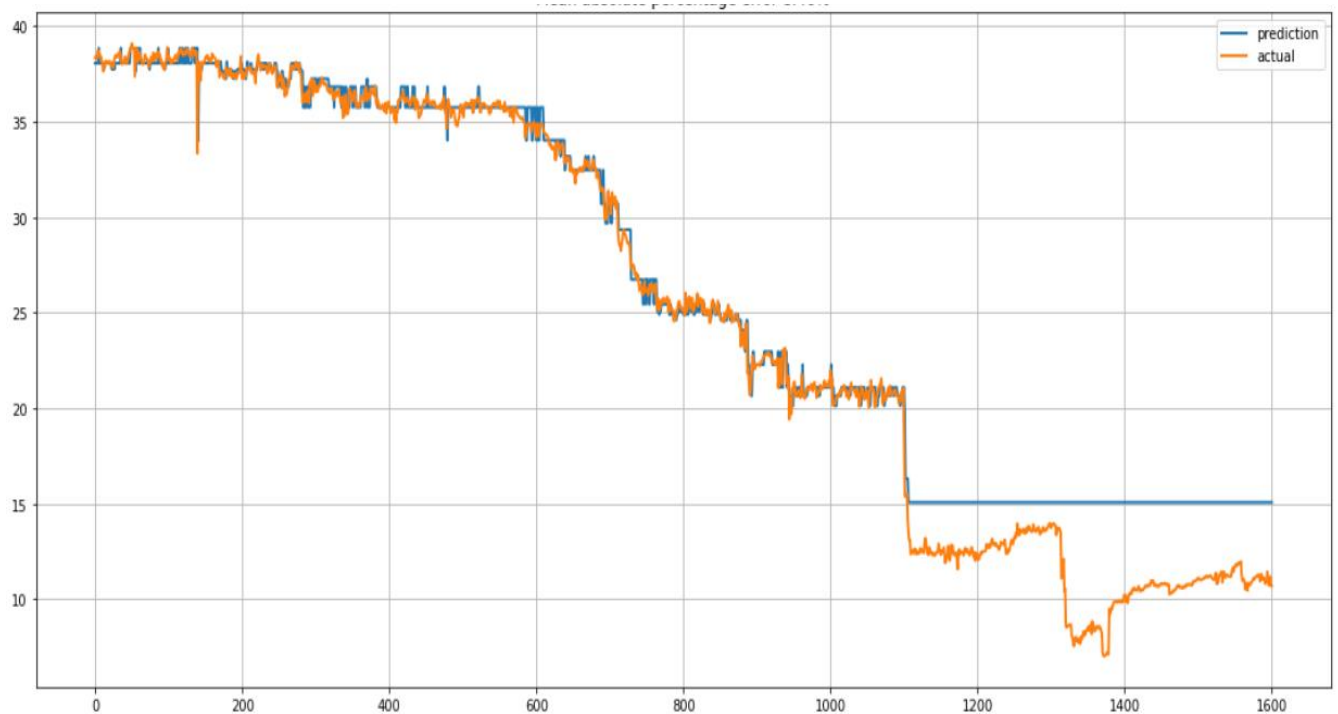
XG BOOST:

The mean squared error is: 5.29305347128611

The R 2 is: 1.3979010737059936

The mean absolute error is: 0.9404522203919421

The root_mean square error is: 2.3006637023446324



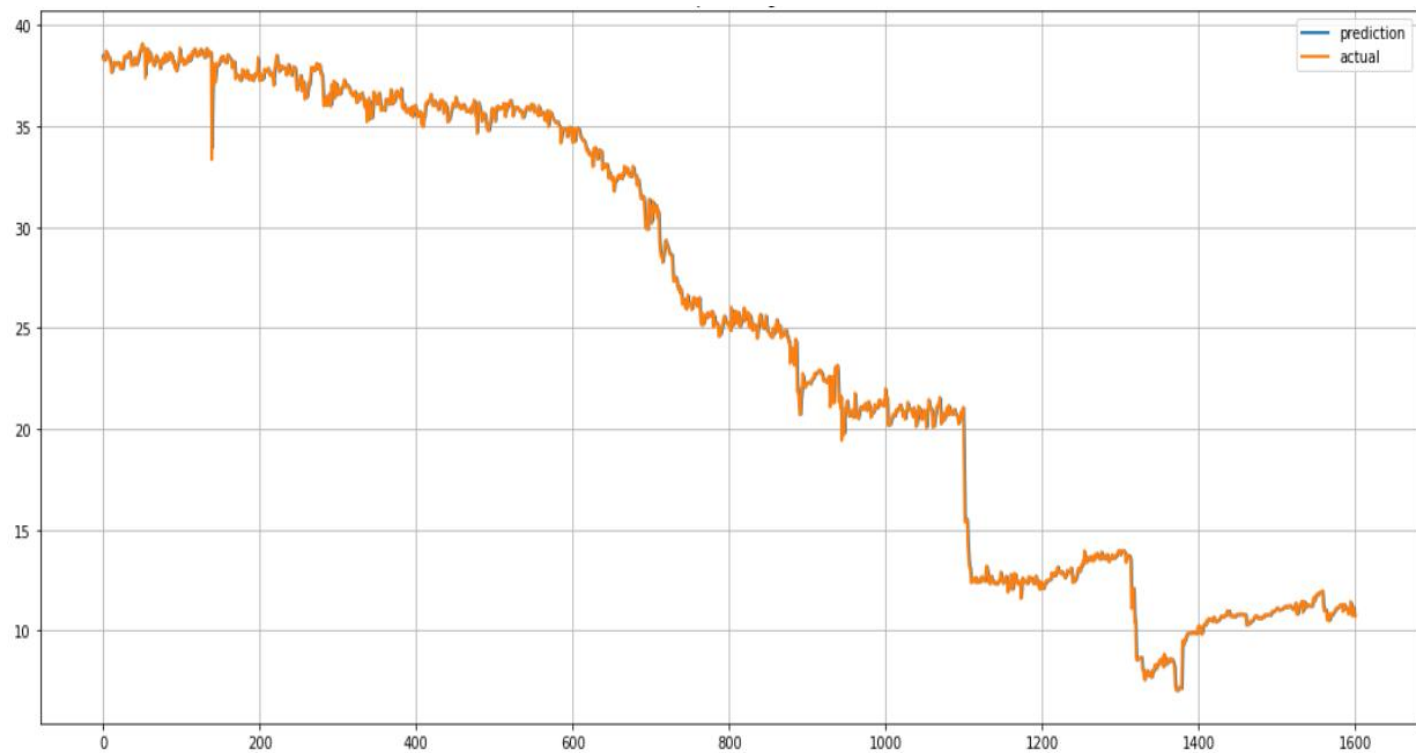
LASSO REGRESSION:

The mean squared error is: 0.1361007746341509

The R 2 is: 0.21377731033022718

The mean absolute error is: 0.9988513967395973

The root_mean square error is: 0.3689183847874092



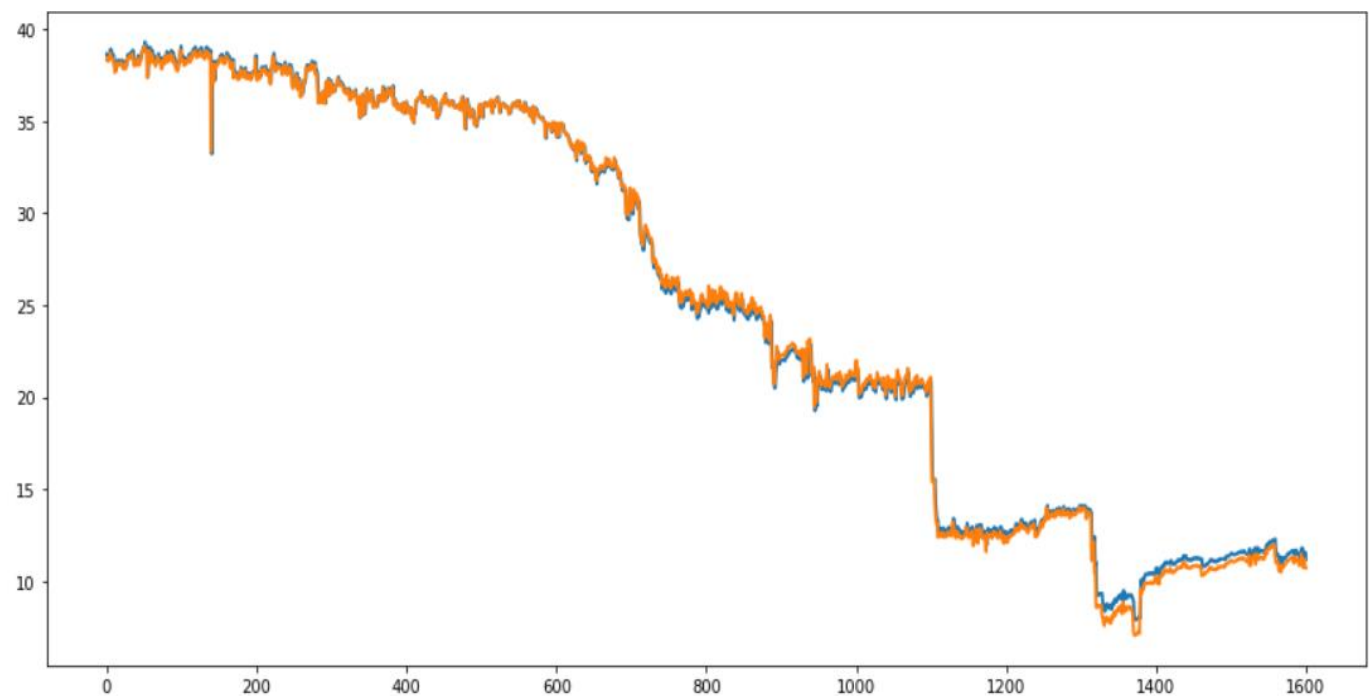
RNN:

The mean squared error is: 0.2108577

The R 2 is: 0.32554892

The mean absolute error is: 0.9982221987833627

The root_mean square error is: 0.45919245



COMPARISION OF MODEL EVALUATION METRICS FOR GAS PRODUCTION:

Models	RMSE	MSE	R ²	MAE
RNN-LSTM	0.459192	0.2108577	0.32554892	0.9982221
Lasso	0.368918	0.1361007	0.21377731	0.9988513
Random Forest	1.997491	3.9899724	1.0860851	0.9586147
XGBoost	2.300663	5.2930534	1.39790107	0.9404522

CONCLUSION:

Time series forecasting (TSF) is the task of predicting future values of a given sequence using historical data. Here TSF is used to predict Total Flow Rate and Oil Production. TSF helps businesses make informed business decisions because it can be based on historical data patterns. TSF is used to compare the accuracy of the oil production with several standard data analysis techniques RNN, Gradient Boosting, XG Boosting, Random Forest, Lasso Regression by measuring RMSE, MSE, MAE, Root Squared, and accuracy from the algorithms and by comparing the results Lasso Regression outweighs the other algorithms by considering low value of RMSE as high accuracy regression.

REFERENCES:

- [1] J.G. De Gooijer, R.J. Hyndman, 25 years of time series forecasting, *Int. J. Forecast.* 22 (3) (2006) 443–473.
- [2] D.S. Poskitt, A.R. Tremayne, The selection and use of linear and bilinear time series models, *Int. J. Forecast.* 2 (1) (1986) 101–114.
- [3] H. Tong, *Non-Linear Time Series: A Dynamical System Approach*, Oxford University Press, 1990.
- [4] R.F. Engle, Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation, *Econometrica* 50 (4) (1982) 987–1007
- [5] G. Zhang, B.E. Patuwo, M.Y. Hu, Forecasting with artificial neural networks: the state of the art, *Int. J. Forecast.* 14 (1998) 35–62.
- [6] M.H. sken, P. Stagge, Recurrent neural networks for time series classification, *Neurocomputing* 50 (2003) 223–235.

- [7] Bayer, J. Simon, Learning Sequence Representations, Technische Universitt Mnchen, 2015.
- [8] Pascanu, Razvan, T. Mikolov, Y. Bengio, On the difficulty of training recurrent neural networks, in Proceedings of the 30th International Conference on Machine Learning (3), volume 28, 2013, pp. 1310–1318.
- [9] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
- [10] I. Sutskever, Training recurrent neural networks, University of Toronto, 2012 Ph.d. thesis.
- [11] M. Längkvist, L. Karlsson, A. Loutfi, A review of unsupervised feature learning and deep learning for time-series modeling, Pattern Recognit. Lett. 42 (2014) 11–24.
- [12] M. Hermans, B. Schrauwen, Training and analyzing deep recurrent neural networks, in: Proceedings of the 26th International Conference on Neural Information Processing Systems NIPS 1, 2013, pp. 190–198.
- [13] R. Pascanu, C. Gulcehre, K. Cho, Y. Bengio, How to construct deep recurrent neural networks, in Proceedings of the Second International Conference on Learning Representations ICLR, 2014.
- [14] P.E. Utgoff, D.J. Straczuzi, Many-layered learning, Neural Comput. 14 (10) (2002) 2497–2529.
- [15] Q. Yang, X. Wu, 10 challenging problems in data mining research, Int. J. Inf. Technol. Decis. Making 5 (2006) 597–604.