Introduction and Literature Survey

The viscosity of a fluid is its opposition to a alteration in the form or movement of adjacent portions relative to each other. Viscosity denotes resistance to flow.

However, in many industrial applications, we are not sure about the viscosity of the fluid and the measurement methods are not very feasible and under budget. To solve this problem, researchers have come out with different methods to predict the instantaneous viscosity of a fluid given some of its physical properties like shear rate, shear stress, strain, composition, temperature, etc.

Machine Learning/Artificial Intelligence models are widely used these days for a variety of regression and classification-based problems. We have tried inculcating these models for the prediction of instantaneous viscosity.

Microalgae Slurry:

Studying about the physical properties of microalgae slurry and how they affect the dynamic viscosity and other properties.

Artificial Neural Network mode was used to obtain a model which can predict the dynamic viscosity of microalgae slurry with a significant amount of accuracy.

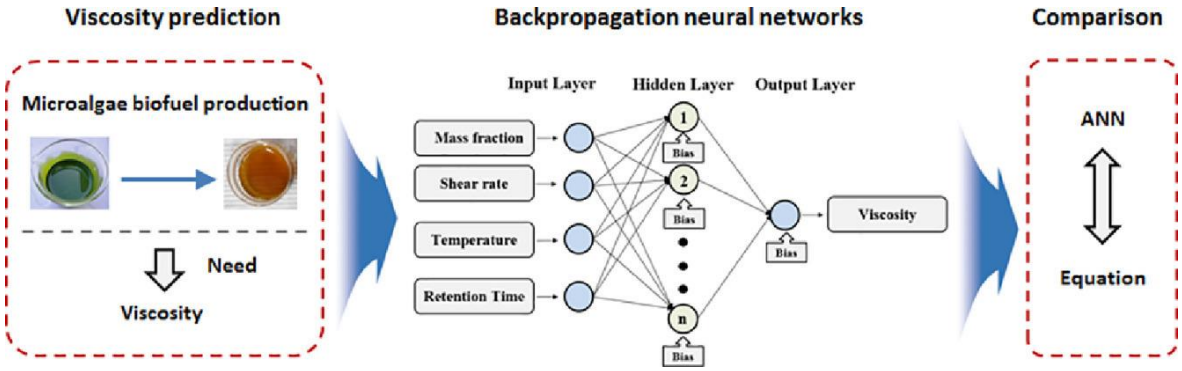


Figure 1 : https://doi.org/10.1016/j.egyai.2021.100053

The hydrothermal hydrolysis process required knowledge of dynamic viscosity. The model estimated the dynamic viscosity using weight fraction, shear rate, temperature, and retention time as input variables.

Nanofluids:

Predicting the dynamic viscosity of graphite nanofluids using Artificial Neural Network.

In this research, graphite particles are chosen to create a nanofluid combination with pure water as the base fluid. Their volumetric ratios in pure water vary between zero to 2%. Two different neural networks are developed here. One with input pairs as Reference water viscosity Vs nano particle concentration and other with input pairs as Temperature Vs nano particle concentration.

Biodiesel fuel

Biodiesel is manufactured using widely available feedstocks such as vegetable oils, animal fats and oils, waste cooking oils, algae, and other oil-based wastes. It possesses diesel-like fuel characteristics that are typically determined through experimentation.

The experimental determination of biodiesel's many attributes is complex and expensive and few laboratories are equipped to conduct these studies.
To address these issues, Artificial Neural Network (ANN) has been identified as a critical tool for assessing biodiesel fuel properties.

A neural network architecture with 5 neurons in the input layer, 2 neurons in the first hidden layer, and 4 neurons in the second hidden layer was used, with neurons in each layer.
The model was created and trained to assess biodiesel's cetane number (CN), flash point (FP), kinematic viscosity (KV), and density based on its fatty acid (FA) content.

Objective

To collect data from various research papers for different types of fuels to create a database and use it to build a suitable machine learning/deep learning model which can predict the instantaneous viscosity with a significant reduction in errors in prediction along with generalizing over a number of different fuels.

To understand and analyse the output for different compositions of fuels and incorporate it in our model.
Code Artificial Neural Network and Random Forest Regressor models for better prediction of instantaneous viscosity.

Approach/Methodology

For the prediction of instantaneous viscosity through Machine Learning/Deep Learning models were designed in this study. Here Artificial Neural Networks and Random Forest models have been used.

Artificial Neural Networks: Neural Networks are a collection of algorithms that attempts to spot trends and extract information from incoming information. A neural network is made up of three layers: an input layer, a hidden layer, and an output layer.

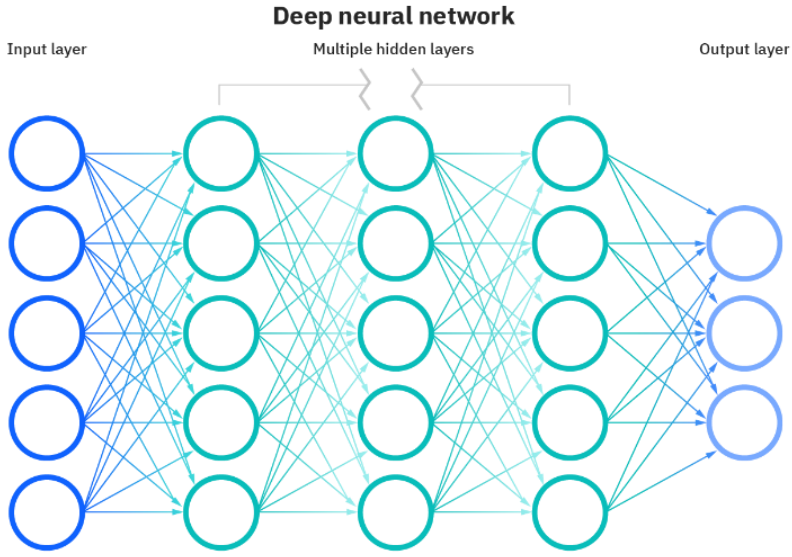


Figure 2: https://1.cms.s81c.com/sites/default/files/2021-01-06/ICLH_Diagram_Batch_01_03-DeepNeuralNetwork-WHITEBG.png

The number of layers and neurons in each layer are varied for optimal prediction of instantaneous viscosity based upon the complexity and size of the data.

Procedure

First, we tried on creating models on datasets of already published research papers.

Code:

A 3-layer Artificial Neural Network consisting of 3, 2 and 2 neurons respectively were used to train the dataset.

#Import necessary libraries

```
import tensorflow as tf
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

#Reading and printing out the dataset

```
df = pd.read_csv("Microalgae_slurry.csv")
df.head()
```

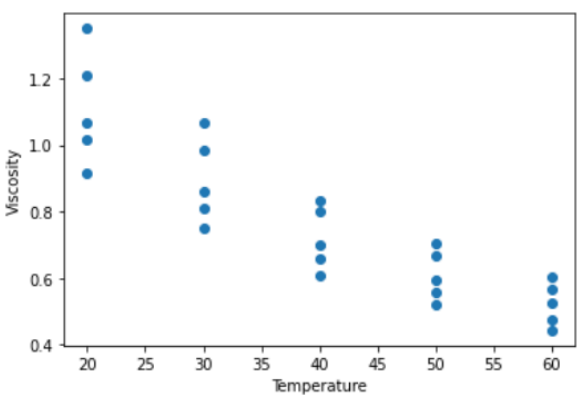
	Volume Concentrattion	Temperature	Time	Viscosity	Density	Volume	Average measured Viscosity
0	0.0	20	114.90	0.919	0.9980	0.917	0.000917
1	0.0	30	94.40	0.755	0.9960	0.752	0.000752
2	0.0	40	76.60	0.613	0.9920	0.608	0.000608
3	0.0	50	66.00	0.528	0.9880	0.521	0.000521
4	0.0	60	56.30	0.450	0.9830	0.443	0.000443

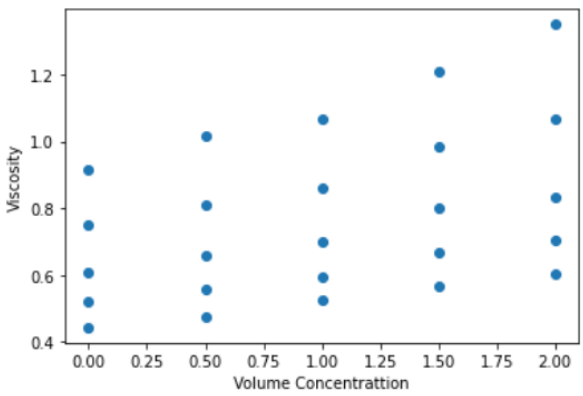
 # Separating the dataset into the input features (i.e., the independent variables) and the target values (dependent variables).

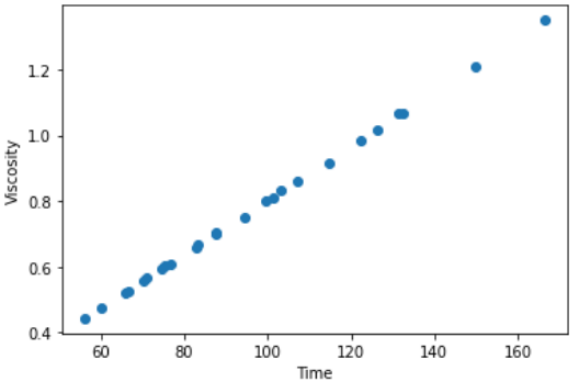
```
X = df.iloc[:, :-1]
Y = 1000*df[["Average measured Viscosity"]]
```

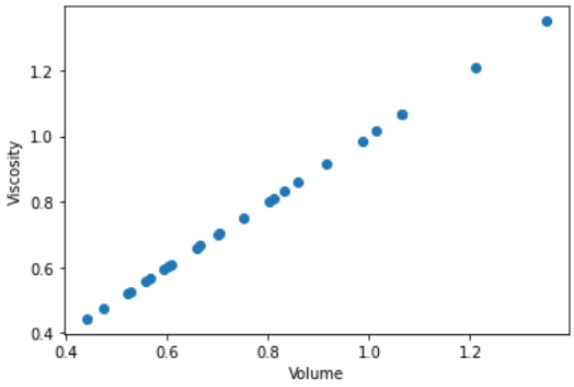
 # Plotting Viscosity values v/s all the input features individually

```
for i in X:
    plt.scatter(df[i], Y)
    plt.xlabel(i)
    plt.ylabel("Viscosity")
    plt.show()
```



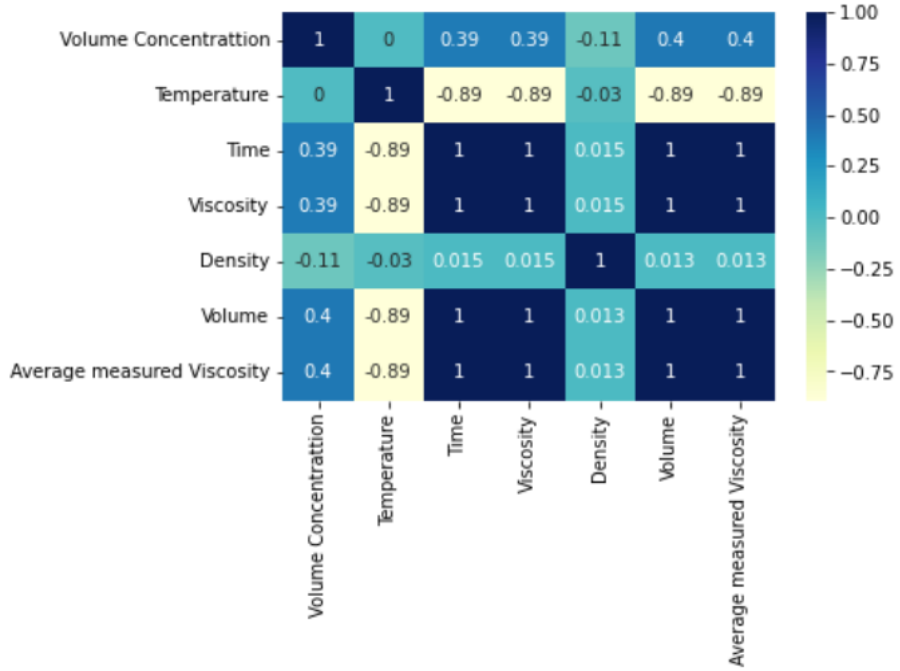






#Observing correlation among all the variables

```
import seaborn as sb
dataplot = sb.heatmap(df.corr(), cmap="YlGnBu", annot=True)
plt.show()
```



#Selecting the input features

Here we are taking in temperature, volume concentration and time in comparison to the research paper where only temperature and volume concentration were taken into accounts.

```
X = df[["Temperature", "Volume Concentrattion", "Time"]]
```

#Data Normalization

```
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
X_scaled = scaler.fit_transform(X)
```

 # Train-test split and training

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X_scaled, Y, test_size=0.1, random_state=2)
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LeakyReLU, PReLU, ELU, ReLU
from tensorflow.keras.layers import Dropout
```

```
classifier = Sequential()
```

```
classifier.add(Dense(units=3, activation="relu"))
classifier.add(Dense(units=2, activation="relu"))
classifier.add(Dense(units=2, activation="relu"))
classifier.add(Dense(units=1, activation="relu"))
```

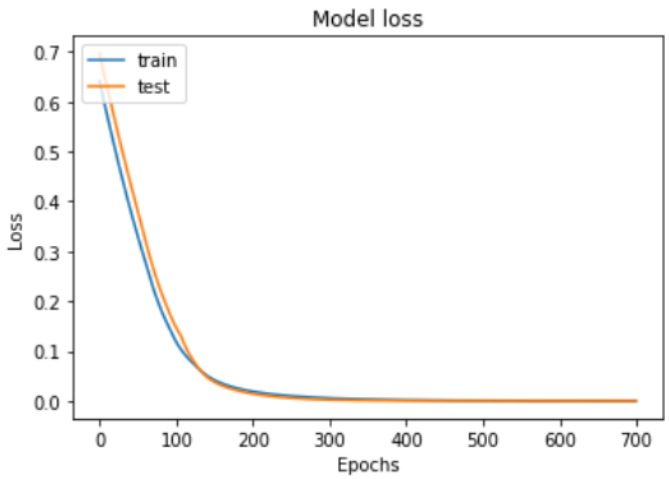
```
classifier.compile(optimizer="adam", loss=tf.keras.losses.MeanSquaredError())
```

```
model_history = classifier.fit(x_train, y_train, validation_split=0.1, epochs=700)
```

```
Epoch 1/700
1/1 [==============================] - 1s 1s/step - loss: 0.6408 - val_loss: 0.6967
Epoch 2/700
1/1 [==============================] - 0s 53ms/step - loss: 0.6334 - val_loss: 0.6886
Epoch 3/700
1/1 [==============================] - 0s 59ms/step - loss: 0.6261 - val_loss: 0.6806
Epoch 4/700
1/1 [==============================] - 0s 52ms/step - loss: 0.6188 - val_loss: 0.6727
Epoch 5/700
1/1 [==============================] - 0s 51ms/step - loss: 0.6119 - val_loss: 0.6663
Epoch 6/700
```

#Plotting and testing our model

```
plt.plot(model_history.history["loss"])
plt.plot(model_history.history["val_loss"])
plt.title("Model loss")
plt.ylabel("Loss")
plt.xlabel("Epochs")
plt.legend(["train", "test"], loc = "upper left")
plt.show()
```




```
y_pred = classifier.predict(x_test)
```

```
y_pred
```

```
array([[0.5167017 ],
       [0.91710436],
       [0.8028315311, dtype=float32)
```

```
np.abs(((y_test - y_pred)*100)/y_test)
```

	Average measured Viscosity
14	1.954137
0	0.011381
17	0.228656

We were able to better the model and reduce the average error to 0.73% in comparison to the research paper where the average error was 0.915%.

Secondly, we moved on to collect a number of datasets from multiple research papers thereby creating a database. However, the discrepancies in the features taken for each paper i.e., different features were taken for different research papers., made the creation and training of a model nearly impossible.

We then started training on GBMg, GBAl and GBTi where the particle concentration was varied (10%, 20% and 30%).

Individual datasets were trained first (like GBMg10) to test our hypothesis and then we introduced a new feature of composition of the particles to develop the model thus creating three models respectively for GBMg, GBAl and GBTi.

We were able to predict instantaneous viscosities with mean percentage error less than 5% error and thus completing our aim.

We then moved on to the extension of our aim i.e., generalizing our model to inculcate all the fuels based upon the density of the particles that were used with the base fuel as JetA1 and stabilizer as Thixatrol.

Here we have used a Machine Learning model: Random Forest Regressor

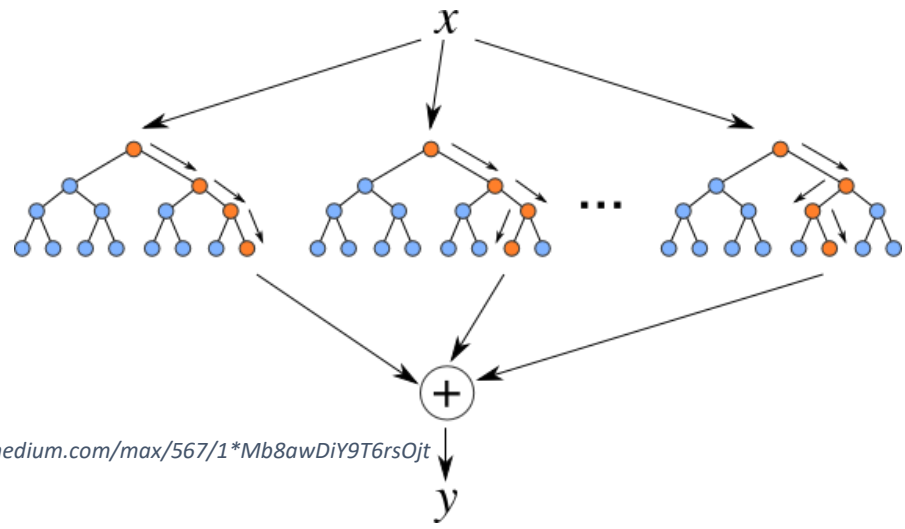


*Figure 3: https://miro.medium.com/max/567/1*Mb8awDiY9T6rsOjtNTRcIg.png*

Here DensityX represents the density of the particle in the fuel.

Performing similar Data Preprocessing as done previously:

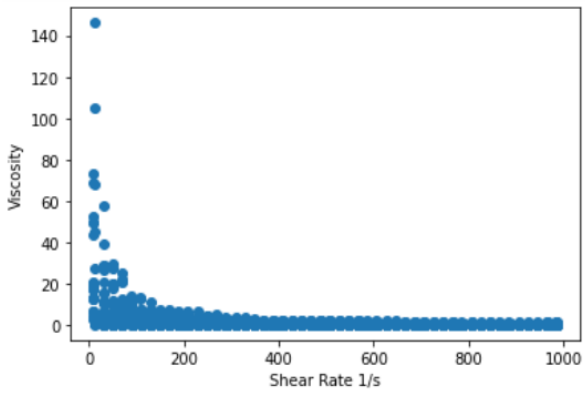
#Performing Statistical Analysis of the data

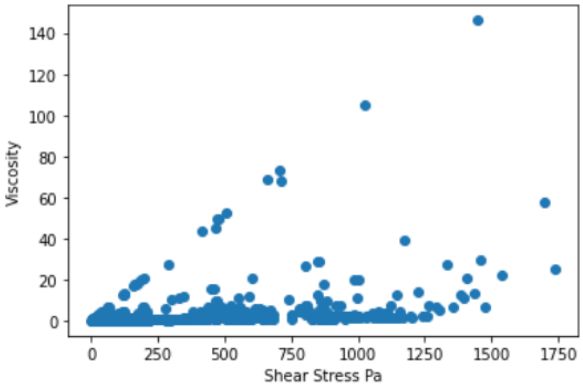
```
df.describe()
```

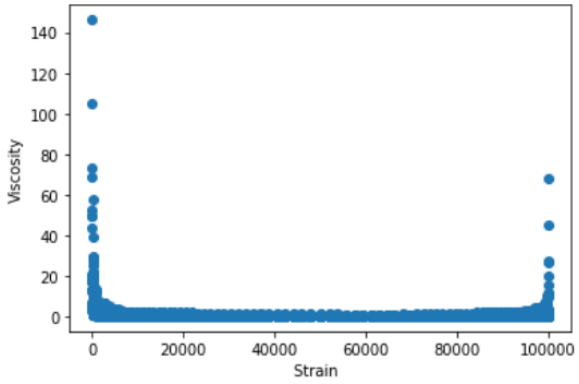
	DensityX	Composition	Time s	Shear Rate 1/s	Shear Stress Pa	Instantaneous Viscosity Pas	Strain
count	2300.000000	2300.000000	2300.000000	2300.000000	2300.000000	2300.000000	2300.000000
mean	2.957478	20.000000	100.056567	497.757265	162.484878	1.132073	50514.812500
std	1.098812	7.803592	57.782246	287.382027	253.415903	5.767858	36545.155785
min	1.740000	10.000000	1.004000	9.441000	2.085000	0.012820	19.010000
25%	1.740000	10.000000	50.515000	249.000000	42.520000	0.081392	13250.000000
50%	2.700000	20.000000	100.015000	497.750000	61.660000	0.150800	51000.000000
75%	4.506000	30.000000	149.600000	746.400000	124.325000	0.412750	87825.000000
max	4.506000	30.000000	199.100000	986.200000	1736.000000	146.200000	100000.000000

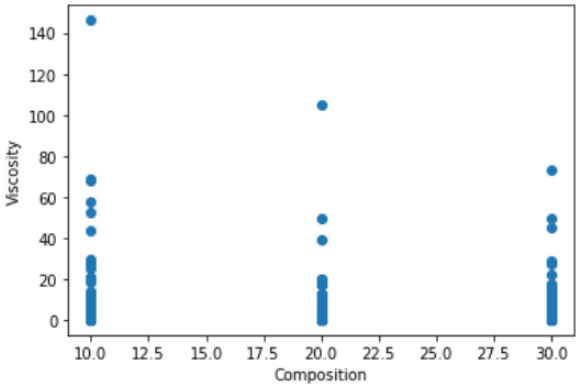
```
X = df[["Shear Rate 1/s", "Shear Stress Pa", "Strain  ", "Composition", "DensityX", "Time s"]]
Y = df[["Instantaneous Viscosity Pas"]]
```

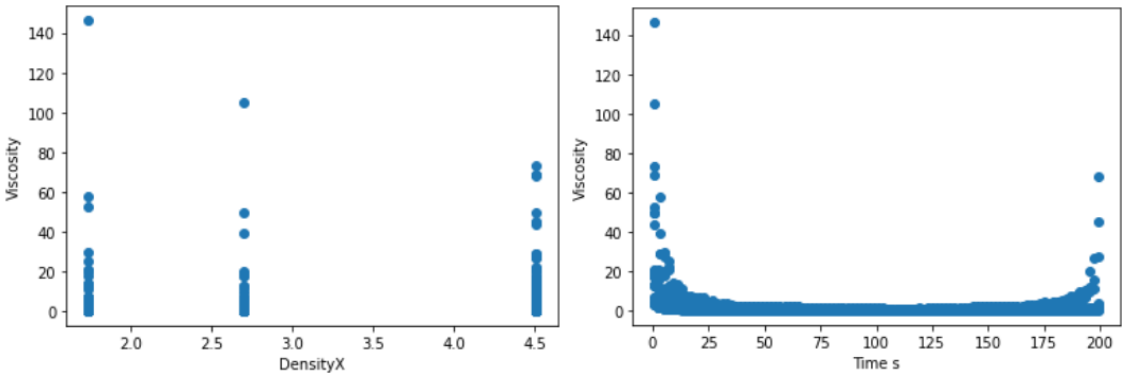
```
for i in X:
    plt.scatter(df[i], Y)
    plt.xlabel(i)
    plt.ylabel("Viscosity")
    plt.show()
```



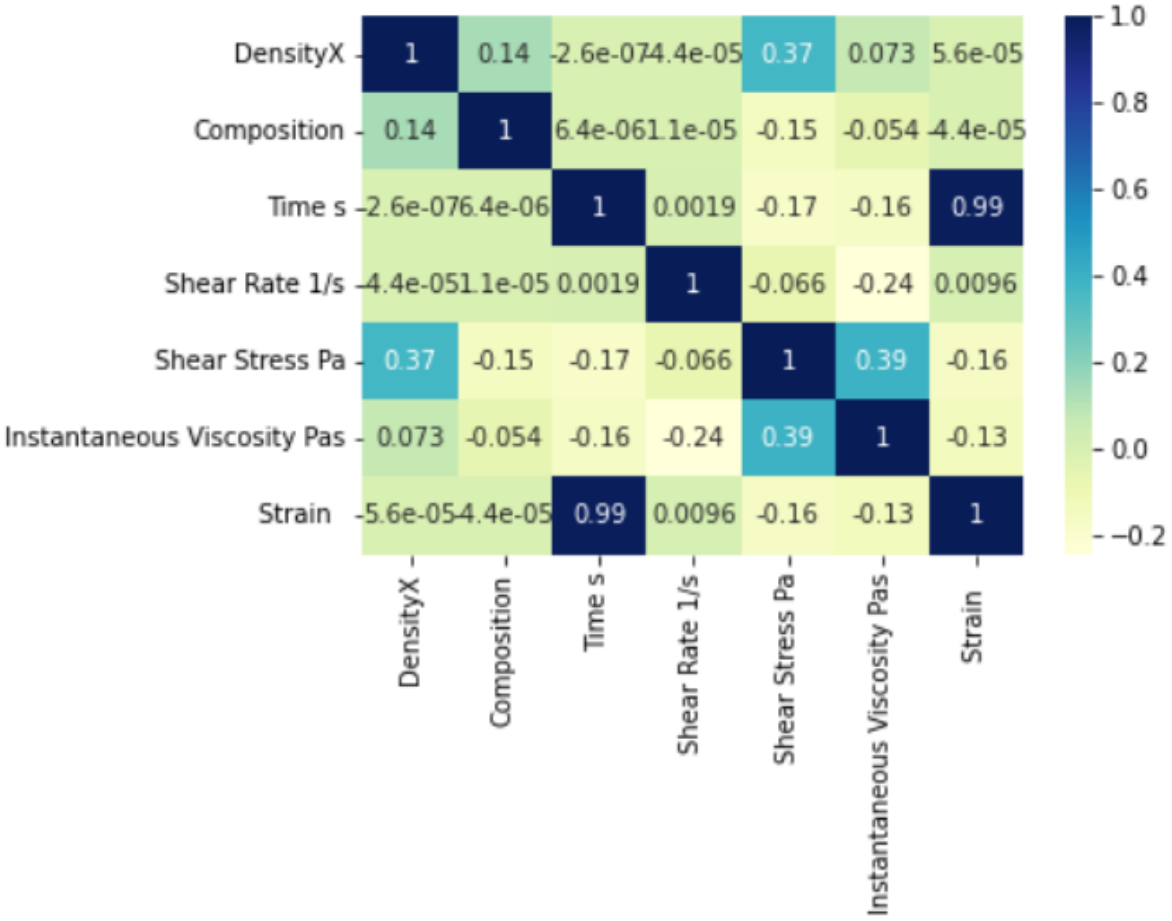









```
import seaborn as sb
dataplot = sb.heatmap(df.corr(), cmap="YlGnBu", annot=True)
plt.show()
```



```
from sklearn.ensemble import RandomForestRegressor
regr = RandomForestRegressor(n_estimators = 200, random_state = 42)
regr.fit(x_train, y_train)
```

```
C:\Users\Main\AppData\Local\Temp\ipykernel_24896\2442269255.py:3: Dat
array was expected. Please change the shape of y to (n_samples,), for
  regr.fit(x_train, y_train)
```

```
RandomForestRegressor(n_estimators=200, random_state=42)
```

```
y_test_pred = regr.predict(x_test)
y_test_pred = y_test_pred.reshape(230,1)
```

```
error = (np.sum(abs((y_test - y_test_pred)/y_test)))/2.3
error
```

```
Instantaneous Viscosity Pas    3.301499
dtype: float64
```

The Mean Absolute Percentage Error was 3.3% while generalizing the model over all the fuels with varying compositions.

Results and Discussion

In the implementation of a research paper for predicting the dynamic viscosity of graphite nanofluids using Artificial Neural Networks we were able to achieve better results, by taking into account an extra feature and changing some hyperparameters of our model, of error less than 0.73%.

In the individual modelling of all fuels, we were able to achieve a Mean Absolute Error Percentage of less than 5% through the use of Random Forest Regressor.

In the extension of our aim i.e., generalizing the prediction of viscosity for all fuel categories we were also able to achieve results well within the error range of 5%. However, when we tried predicting the fuels having particles of density that is not close within the trained model, we were generating error of 20%.

Conclusions and Recommendation for future work

Generalization of model over all the fuel particles is still an area which needs improvement. Since the data available was for only three fuels with varying particles of Mg, Al and Ti, were not able to predict viscosity of fuels where the particle density where well outside the range.

Creating an extended dataset including few more fuels will help in achieving much better results and predicting over a wide range of particles.