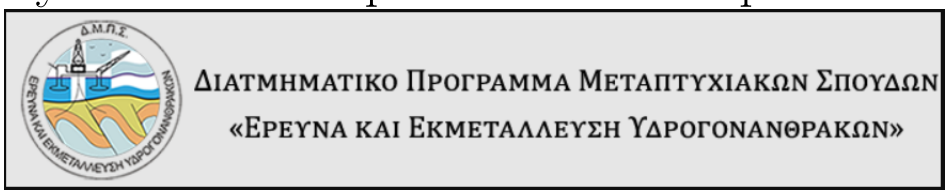
Machine learning models to predict reservoir fluid properties

Hydrocarbon Exploration and Exploitation



HGD 5Y: Master Dissertation Project

Sofianos-Panagiotis Fotias

Supervisor: Vassilis Gaganis (Assistant Professor)

Thesis Examination Committee:
Vassillis Gaganis (Supervisor)
Georgakopoulos Andreas (Professor)
Benardos Andreas (Professor)

Collaborating Institutions:
Aristotle University of Thessaloniki - School of Geology
National Technical University of Athens - School of Mining Engineering and Metallurgy
Democritus University of Thrace - School of Economics
National and Kapodistian University of Athens - School of Geology and Geoenvironment

Contents

Abstract

This project's goal was to look at the prediction of the undersaturated oil viscosity of various resevoir fluids. Our first action was to determine whether or not existing correlations could actually predict the undersaturated oil viscosity at pressures above the bubble point. To do this we used 18 correlations from the literature and used statistics such as average absolute error and graphical error analysis, in order to find the ones the better fit our dataset. Our next goal was to build our own models using machine learning algorithms. To do this, we first had to process our data and then, we fitted those algorithms for a reduced set of inputs (P,Pb,μ_{ob}) plus the full set of inputs from our PVT report (P,Pb,μ_{ob},GOR,API,T). We then shuffled our data 100 times and fitted every algorithm again in order to make a final comparison between our models and the literature. In the end we managed to not only fit algorithms that perform well, but actually tune them so as to perform better than the ones existing in the literature.

1 Introduction

Viscosity is defined as the quantity that describes a fluid's resistance to flow. Fluids resist the relative motion of immersed objects through them as well as to the motion of layers with differing velocities within them. Dynamic viscosity is defined as the ratio of the shearing stress (F/A) to the velocity gradient $(\Delta\nu_x/\Delta y)$.

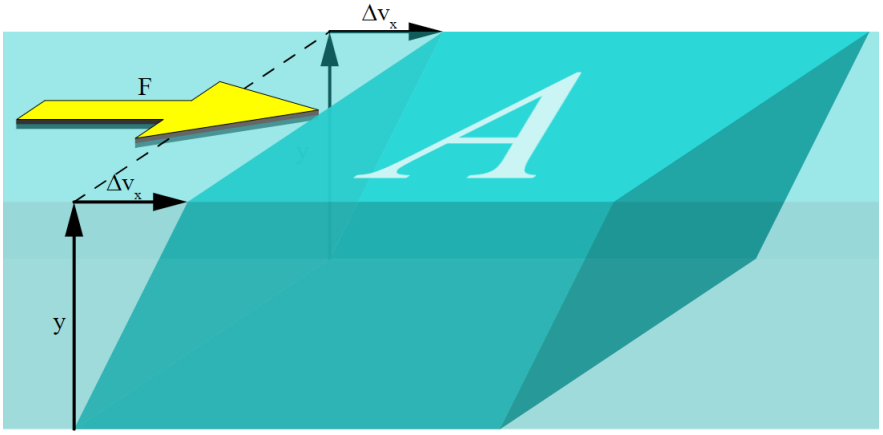


Figure 1: dynamic viscosity

$$\eta = \frac{(F/A)}{(\Delta\nu_x/\Delta y)}$$

This is called Newton's equation and it states that the shear of a fluid is directly proportional to the force applied to it and inversely proportional to its viscosity.

There is a second quantity called viscosity. It is called kinematic viscosity and is the ration of the viscosity of a fluid to its density.

$$\nu = \frac{\eta}{\rho}$$

It is a measure of the resistive flow of a fluid under the influence of gravity.

The fluids that obey the equations above with constant viscosity are called Newtonian fluids. There are also the non-newtonian ones. The viscosity of those fluids is a function of some mechanical variable like shear stress or time.

Oil is generally considered to be a Newtonian fluid. However, its viscosity varies with changes in pressure, density and temperature, but not in such an extent as to be considered non-newtonian.

The term undersaturated oil viscosity is used to define the viscosity of the oil for pressures above the bubble point. Bubble point pressure is the point where the first trace of gas forms. Below the bubble point, the reservoir fluid enters the multi-phase region and many properties among viscosity start behaving in a different manner. In this work, only the undersaturated part of the fluid, will be examined. What this means regarding viscosity, is that as pressure drops, viscosity drops as well,

as long as pressure stays above the bubble point. This is a general trend among all undersaturated reservoir fluids, no matter what the initial pressure, viscosity, gravity or other properties are. Viscosity is a very important fluid property, since its measurement or even a good prediction is essential in many calculations in the field of reservoir and production engineering. It is needed for estimating reserves (OOIP), planning for secondary and tertiary recovery methods, calculating flowrate etc. (Kulchanyavivat, 2005)

Viscosity is also needed to determine the pressure drop associated with flow. Darcy's known equation for cylindrical (radial) flow through porous media is presented below.

$$P - P_{wf} = \frac{q_o \mu_o B_o \left[\ln\left(\frac{r_e}{r_w}\right) - 0.75 + S \right]}{7.0815 \cdot 10^{-3} kh}$$

For undersaturated reservoirs, temperature and composition are constant and viscosity changes only with pressure change. Viscosity changes from 5% to 35%/1000 psi, oil formation volume factor (B_o) changes from 0.5% to 2.8%/1,000 psi. This indicates that pressure change is impacted directly and pretty significantly from viscosity drop.

Fluid flow in pipes needs to be calculated as well. It is calculated as a combination of the continuity equation for the conservation of mass and the equation of momentum conservation, where the rate of momentum difference plus the momentum accumulation in a pipe segment must equal the sum of all forces on the fluids. (Mukherjee & Brill, 1999)

Fluid flow in pipes is given by:

$$\frac{\Delta P}{\Delta L} = \frac{1}{144} \left[\rho_o \cos(\theta) + \frac{f \rho_o \nu^2}{2gd} + \frac{\rho_o \nu \Delta \nu}{g \Delta L} \right]$$

where ρ_o is the oil density,ν is the viscosity, θ is the angle between the starting and ending points of a small pipe segment ΔL and the gravitational direction. The first three parts of the equation describe the hydrostatic, frictional and kinetic energy losses in the system, respectively. The frictional factor f, is defined by the Moody Friction factor chart as a function of Reynolds Number and pipe roughness. The Reynolds Number is calculated as:

$$R_e = \frac{1488 \cdot d \cdot \nu \cdot \rho}{\mu}$$

where d is the pipe diameter. For laminar flow at Reynolds numbers less than 2,000, the friction factor is defined as:

$$f = \frac{64}{R_e}$$

Once again, pressure drop is directly proportional to changes in viscosity and density. (Bergman & Sutton, 2006)

Viscosity of oil is usually given by the laboratory (PVT report) and/or correlating equations. The problem that arises from a PVT report is that the time and cost it requires is big. Numerical correlations are frequently used when PVT reports are not available, in order to provide an idea of what the viscosity of the reservoir oil is. These correlations use many properties as inputs, however the most important ones are pressure drop and bubble point viscosity. Others include GOR, API gravity and dead oil viscosity.

It is important to clarify here that for the next chapters Pressure (P) and bubble point pressure (Pb) will be measured in Psi, GOR may be written as Rsob (Rs at the bubble point) and it will be given in scf/STB units, μ_o will be the same as μ and it indicates undersaturated oil viscosity measured in centipoise (cp), while bubble point viscosity (μ_{ob}) and dead oil viscosity (μ_{od}) are also be measured in cp. API and γ_{API} refer to the API gravity and are the same thing and finally temperature (T or Tr) is given in $^{\circ}F$.

This work consists of two parts. The first part was simply reviewing literature correlations on a completely new dataset and extracting various conclusions from it. Then, there was an effort for machine learning regression models to be developed on this dataset, for the prediction of undersaturated oil viscosity. Several of the most popular methods were utilized but first manual feature engineering had to be performed. This task was not trivial, engineering and physics concepts had to be translated into data driven for machine learning algorithms to work in an optimal way. Finally, the methods developed in this work were compared to the literature ones extensively.

2 Literature review

In this section several of the most important correlations that are used to predict undersaturated oil viscosity will be presented. There are of course many more available in the literature, but those fourteen were chosen due to their historic significance and/or popularity. It is by no means implied that the dozens that were not included here are inferior in any way. The methods chosen, either use the pressure ratio or pressure differential as a primary correlating parameter along with bubble point viscosity. Some correlations use additional PVT properties such as solution GOR, API gravity and dead oil viscosity.

2.1 Beal (1946)

In 1946 Beal correlated 52 viscosity observations taken from 26 crude oil samples representing 20 individual oil pools, 11 of which are in California. These observations include viscosity at the bubble point and viscosity at some higher pressure. It was indicated that the viscosity increase was greater when the absolute bubble-point viscosity increased. The viscosity of all samples was noted to increase with pressure above the bubble point.
Beal's correlating mathematical formula is the following:

$$\mu_o = \mu_{ob} + 10^{-5} \cdot (P - P_b) \left(2.4 \cdot \mu_{ob}^{1.6} + 3.8 \cdot \mu_{ob}^{0.56}\right)$$

Beal's prediction of the viscosity in his 26 undersaturated samples showed an average error of 2.7%. The range of viscosity values measured in his experiment were (0.142 to 127 cp) at the bubble point and (0.16 to 315 cp) for the undersaturated point. (Beal, 1946)(M. Standing & Katz, 1981)

2.2 Kouzel (1965)

Kouzel used Barus' equation and correlated α with bubble point viscosity. Kouzel used 95 data points with bubble point viscosity ranging from 1.22 to 134 cp and undersaturated viscosity from 1.78 to 202 cp. bubble point pressure range was just 14.7 psi and undersaturated pressure range was 423 to 6,015 psi. The viscosity-pressure coefficient is calculated from the following equation.

$$\alpha = 5.50318 \cdot 10^{-5} + 3.77163 \cdot 10^{-5} \mu_{ob}^{0.278}$$

(Kouzel, 1965)

We can also find an API modified Kouzel correlation in the literature

$$\alpha = -2.34864 \cdot 10^{-5} + 9.30705 \cdot 10^{-5} \mu_{ob}^{0.181}$$

(API, 1997)

2.3 Vazquez and Beggs (1976)

Vazquez and Beggs considered that many correlations used up to that point were developed many years ago from limited data and were being used beyond the range

for which they were intended to. They gathered more than 600 laboratory PVT analyses from fields all over the world and included more than 6,000 measurements. The data exhibited a wide range of oil properties. They used regression analysis techniques to correlate the laboratory data. Undersaturated oil viscosity was correlated as a function of bubble point viscosity, pressure and bubble point pressure. The viscosity of the oil at Pb was obtained from the Beggs and Robinson correlation.

$$\mu_o = \mu_{ob} \left(\frac{P}{P_b} \right)^m$$

where

$$m = C_1 \cdot P^{C_2} \cdot \exp(C_3 + C_4 P)$$

and $C_1 = 2.6$, $C_2 = 1.187$, $C_3 = -11.513$ and $C_4 = -8.98 \cdot 10^{-5}$

(Vazquez & Beggs, 1980)

2.4 Labedi (1982)

Labedi in his 1982 PhD thesis collected a large number of PVT analyses on the oil and gas reservoirs of Libya, Nigeria and Angola. He used multiple linear regression techniques and tested many possible mathematical models and also studied the effect of each independent variable in order to produce a mathematical method of predicting the dependent variable with the least error and as a function of easily measurable field data.
He noted that for all samples there exists a linear relationship between the oil viscosity and the reservoir pressure above the bubble point. He wrote,

$$\begin{aligned} &\mu_o = \mu_{ob} + m\,(P - Pb) \Rightarrow \\ &\mu_o - \mu_{ob} = m \cdot P_b \left(\frac{P}{P_b} - 1 \right) \Rightarrow \\ &\Delta\mu = M_{\mu\,a} \left(\frac{P}{P_b} - 1 \right) = -M_{\mu\,a} \left(1 - \frac{P}{P_b} \right) \end{aligned}$$

This function is generally a straight line with an intercept equal to zero. He obtained the following relationships for estimating the slope.

- Libyan Crude

$$M_{\mu\,a} = 10^{-2.488} \cdot \mu_{od}^{0.9036} \cdot P_b^{0.6151} / 10^{0.01976 \cdot \gamma_{API}}$$

- Nigerian and Angolan crudes

$$M_{\mu\,a} = 0.0483 \cdot \mu_{od}^{0.7374}$$

For this estimation 91 data points were used for Libyan crude and 31 for Nigerian. Bubble point viscosity range was 0.115 to 3.72 cp for Libyan and 0.098 to 10.9 cp for Nigerian crudes. Bubble point pressure range was 60-6,358 psi for Libyan and 715-4,794 psi for Nigerian crudes.

(Labedi, 1982)

2.5 Khan (1987)

In this study viscosity data for 75 samples taken from 62 Saudi Arabian oil reservoirs were utilized. A total of 150 data points were used for bubble point oil viscosity and 1,503 for oil viscosity above the bubble point. Bubble point oil viscosity range was 0.13-17.9 cp. Viscosity above the bubble point had a range of 0.13-71 cp. Bubble point pressure was between 107 and 4,315 psi. GOR 24-1,901 SCF/STB, API 14.3-44.6, and finally pressure was between 14.7(including saturated measurements) to 5,015 psi. Viscosity correlations were obtained by nonlinear multiple regression analysis utilizing two regression criteria: common least squares and least absolutes. The correlation is the following:

$$\mu_o = \mu_{ob} \exp(9.6 \cdot 10^{-5} (P - Pb))$$

They also used Barus' relation but instead of a function α was determined as a constant. (Khan, Al-Marhoun, Duffuaa, & Abu-Khamsin, 1987)

2.6 Al-Khafaji (1987)

In 1987 Al-Khafaji, Abdul-Majeed, and Hassoon developed correlations in order to predict undersaturated oil viscosity by using 210 oil samples from the Middle East. Their new equation was created as a function of API gravity, bubble point pressure and reservoir pressure. Bubble point viscosity range was 0.093-7.139 cp. The equation is as follows:

$$\mu_o = \mu_{ob} + 10^F$$

$$F = -0.3806 - 0.1845 \cdot \gamma_{API} + 0.004034 \cdot \gamma_{API}^2 - 3.716 \cdot 10^{-5} \cdot \gamma_{API}^3 + 1.11 \log(0.07031(P - P_b))$$

(Al-Khafaji, Abdul-Majeed, Hassoon, et al., 1987)

2.7 Abdul-Majeed (1990)

In 1990 Abdul-Majeed,Kattan and Salman developed a new equation using 253 experimentally obtained oil viscosities on 41 different oil samples from North America and Middle East. Undersaturated viscosity range was 0.096-28.5 cp and bubble point viscosity range was 0.093-20.5 cp. Bubble point pressure ranges from 498 to 4,864 psi. This correlation was derived from plotting ΔP vs $\Delta\mu$ on a log-log paper. The plot revealed a series of straight lines of a constant slope. The interecepts of the resulting lines could be given as a function of stock tank gravity and solution GOR at the bubble point. The equation has the following form:

$$\mu_o = \mu_{ob} + 10^{G - 5.2106 + 1.11 \cdot \log(6.894757(P - P_b))}$$

$$G = 1.9311 - 0.89941 \ln(R_s) - 0.001194 \cdot \gamma_{API}^2 + 9.2545 \cdot 10^{-3} \cdot \gamma_{API} \cdot \ln(R_s)$$

(Abdul-Majeed, Kattan, & Salman, 1990)

2.8 Petrosky (1990)

In 1990 Petrosky presented an updated review of the published correlations and provided his own based on 81 laboratory PVT analyses and 404 total data points from the Gulf of Mexico. Bubble point viscosity ranged from 0.211 to 3.546 cp and undersaturated visvosity ranged from 0.22 to 4.09 cp. Bubble point pressure was between 1,574 to 9,552 psi and undersaturated from 1,600 to 10,250 psi. His formula is the following:

$$\mu_o = \mu_{ob} + 1.3449 \cdot 10^{-3}(P - P_b) \cdot 10^{X_2}$$

$$X_1 = \log(\mu_{ob})$$

$$X_2 = -1.0146 + 1.3322 \cdot X_1 - 0.4876 \cdot X_1^2 - 1.15036 \cdot X_1^3$$

(Petrosky, 1990)

2.9 Kartoatmodjo and Schmidt (1991)

In 1991, Kartoatmodjo and Schmidt used many PVT reports from multiple geographical locations such as the Southeast Asia, Latin America, the Middle East, and North America,totaling 3,588 points, to modify Standing's correlation (M. B. Standing, 1977) for undersaturated oil viscosity. Bubble point viscosity ranged from 0.168 to 184.86 cp while the undersaturated one reached the value of 517.03 cp. bubble point pressure range was from 25 to 4,775 psi and the maximum pressure in their dataset reached 6,015 psi. The equation is as follows:

$$\mu_o = 1.00081\mu_{ob} + 1.127 \cdot 10^{-3}(P - P_b)\left(-6.517 \cdot 10^{-3} \cdot \mu_{ob}^{1.8148} + 0.038 \cdot \mu_{ob}^{1.59}\right)$$

(Kartoatmodjo, Schmidt, et al., 1991)

2.10 Orbey and Sandler (1993)

Orbey and Sandler presented a sequence of empirical models for calculating the viscosity of hydrocarbons and their mixtures (including these with carbon dioxide) at both atmospheric and high pressures over a wide range of temperatures. They used 377 data points in pressure range of 740 to 14,504 psi, bubble point viscosity from 0.217 to 3.1 cp and undersaturated viscosity from 0.225 to 7.3 cp.

$$\mu_o = \mu_{ob} \cdot \exp(\alpha(P - P_b))$$

- Parriffinic hydrcarbons

$$\alpha = 6.76 \cdot 10^{-5}$$

- akylbenzes and cyclic hydrocarbons

$$\alpha = 7.24 \cdot 10^{-5}$$

- average

$$\alpha = 6.89 \cdot 10^{-5}$$

(Orbey & Sandler, 1993)

2.11 De Ghetto (1994)

In 1994, De Ghetto, Paone, and Villa introduced a new strategy to produce oil viscosity equations based on 4 different classes of API gravity. The best correlations both for each class and for the whole range of API gravity were evaluated using 195 oil samples and a total of 3,700 measured data. The functional forms of the correlations that gave the best results for undersaturated oil viscosity have been used for finding a better correlation with average errors reduced by 5-10%. The authors mentioned that "the Non-Newtonian behavior of a highly viscous fluid could affect the reliability of laboratory measurement and the performance of viscosity correlation equations". The modified viscosity correlation equations are provided below:

- Extra heavy oil $\gamma_{API} \leq 10$ (Labedi)

$$M_{\mu\, a} = 10^{-2.19} \cdot \mu_{od}^{1.055} \cdot P_b^{0.3132} / 10^{0.0099 \cdot \gamma_{API}}$$

- Heavy oil $10 \leq \gamma_{API} \leq 22.3$ (Kartoatmodjo)

$$\mu_o = 0.9886\mu_{ob} + 2.763 \cdot 10^{-3}(P - P_b)\left(-11.53 \cdot 10^{-3} \cdot \mu_{ob}^{1.7933} + 0.0316 \cdot \mu_{ob}^{1.5939}\right)$$

- Medium oil $22.3 \leq \gamma_{API} \leq 31.1$ (Labedi)

$$M_{\mu\, a} = 10^{-3.8055} \cdot \mu_{od}^{1.4131} \cdot P_b^{0.6957} / 10^{0.00288 \cdot \gamma_{API}}$$

- Agip (Labedi)

$$M_{\mu\, a} = 10^{-1.9} \cdot \mu_{od}^{0.7423} \cdot P_b^{0.5026} / 10^{0.0243 \cdot \gamma_{API}}$$

(De Ghetto & Villa, 1994)

2.12 Almehaideb (1997)

In 1997 Almehaideb used data sets form over 15 different reservoirs in the UAE and generated PVT correlations. He claimed that the improvement in the accuracy of his correlation compared to the one of Vazquez and Beggs ,which is similar in form, may be due to the fact that solution GOR in addition to pressure was included. His crudes had the following ranges:

$$30.9 \leq \gamma_{API} \leq 48.6$$
$$190 \leq T(^\circ F) \leq 306$$
$$501 \leq P_b \leq 4822$$
$$128 \leq R_s \leq 3871$$

As already mentioned he used Beggs and Robinson correlation which utilizes the ratio of Pressure to bubble point pressure. Almehaideb found the exponent to be the following:

$$m = 0.134819 + 1.94345 \cdot 10^{-4} \cdot R_s - 1.93106 \cdot 10^{-9} \cdot R_s^2$$

(Almehaideb, 1997)

2.13 Elsharkawy and Alikhan (1999)

In 1999 Elsharkawy and Alikhan developed new empirical models for predicting Middle East oil viscosity. They used 254 crude oil samples. They found that the slopes of the lines ΔP and $\Delta\mu$ are a function of deadoil viscosity and bubble point pressure. Oil viscosity range was 0.2 to 5.7 cp and oil pressure range was 1,287 to 10,000 psi. The model is the following:

$$\mu_o = \mu_{ob} + \frac{10^{-2.0771}(P - P_b)\mu_{od}^{1.19279}}{\mu_{ob}^{0.40712} P_b^{0.7941}}$$

(Elsharkawy & Alikhan, 1999)

2.14 Dindoruk and Christman (2004)

In 2004 these two researchers used more than 100 PVT reports with bubble point pressure range between 926 and 12,230 psia and GOR range from 133 to 3,050 scf/STB. For their correlation the used the Solver tool built in Microsoft Excel. For the purpose of finding the viscosity correlation they used 95 data points. Some more data ranges are as follows:

$$\begin{aligned}
&17.4 \leq \gamma_{API} \leq 40 \\
&121 \leq T(^\circ F) \leq 276 \\
&202 \leq P - P_b \leq 10,140 \\
&0.161 \leq \mu_{ob} \leq 8.7 \\
&0.211 \leq \mu_o \leq 10.6
\end{aligned}$$

Their best regression analysis results were obtained with the following equation

$$\mu_o = \mu_{ob} + \alpha_6(P - P_b)10^A$$

$$A = \alpha_1 + a_2 \log \mu_{ob} + \alpha_3 \log Rs + \alpha_4 \mu_{ob} \log R_s + \alpha_5(P - P_b)$$

where $\alpha_1 = 0.776644115$, $\alpha_2 = 0.987658646$, $\alpha_3 = -0.190564677$, $\alpha_4 = 0.009147711$, $\alpha_5 = -0.000019111$, $\alpha_6 = 0.00006334$

(Dindoruk & Christman, 2004)

2.15 Hossain (2005)

In 2005 Hossain et al evaluated the existing correlations against a dataset of heavy oils ($10 \leq \gamma_{API} \leq 22.3$) and developed a new correlation. They derived their correlation from performing Quasi-Newtonian non linear regression analysis of their dataset with Beal's equation. The dataset's bubble point viscosity range was 3.6 to 360 cp and the undersaturated was 3 to 517. Bubble point pressure range 222 to 1,458 psi and undersaturated pressure ranged from 300 to 5,000 psi.

$$\mu_o = \mu_{ob} + 0.004481(P - P_b) \cdot \left(0.555955\mu_{ob}^{1.068099} - 0.527737\mu_{ob}^{1.063547}\right)$$

(Hossain, Sarica, Zhang, Rhyne, & Greenhill, 2005)

3 Dataset and literature evaluation

3.1 Dataset

The dataset utilized in this work consists of viscosity datasets on more than 500 reservoir fluids from various published sources and inhouse measurements around the world. There are only a few missing values of Temperature, GOR and API gravity. About 100 datasets don't have compositional and density data but that is not important for the purposes of this work. Pressure and viscosity data are available for all fluids in our database.

The dataset has no fluids with inconsistent viscosity shape, which implies that all data points of every fluid are sort of continuous and smooth and there are no abrupt jumps in measured viscosity values for neighboring points. There are also no outliers and every fluid exhibits increasing viscosity with pressure. Two fluids exhibited first few viscosity measurements after the bubble point less than the bubble point viscosity, but it's only marginal and viscosity quickly ascends afterwards. It should also be pointed out that for every fluid there exist many viscosity measurements that go from the bubble point up to a maximum value of pressure, totaling more than 11000 data points.

The aim of this work is not to just review the known correlations with this new dataset, but also for new correlations to be developed using machine learning techniques. For machine learning algorithms to work properly, as many data points as possible need to be introduced to them. Unfortunately most of the fluids in our dataset are Newtonian like, meaning that their bubble point viscosity is less than 10 cp. To be precise, 89% of fluids have bubble point viscosity of less than 10 cp, 7.8% of them have a viscosity value between 10 and 50 cp and only 3.2% have a value greater than 50, ranging from 59 up to 1,760 cp. It is also known that viscosity values that are this high tend to produce oil that is highly viscous and behaves very differently compared to low viscosity oils. This means that for our models to be developed properly the removal of these very heavy fluids was deemed necessary. The literature's correlations' score against these fluids will still be shown, but our models will not be developed, tested and judged against those points.

It's time to for the maximum and minimum values of our dataset's undersaturated properties to be given and also for the histograms to be plotted to show where most values of each property are concentrated. The y axis is given in percentage value.

- For the whole dataset

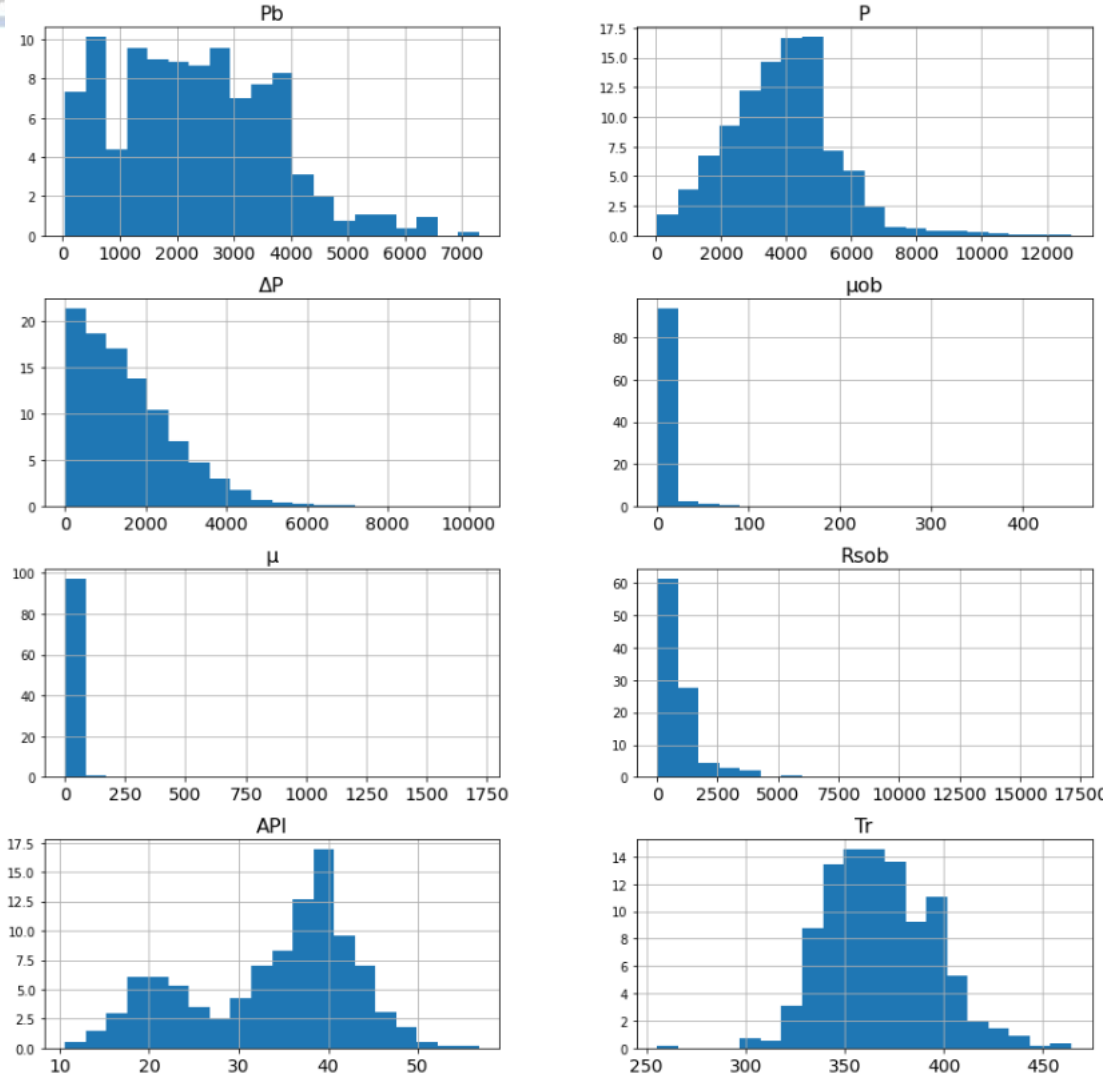


Figure 2: $0 \leq GOR \leq 17,118.6$ $10.58 \leq \gamma_{API} \leq 56.84$ $255.37 \leq T \leq 464.25$ $0.0416 \leq \mu_o \leq 2,455.5$ $0.0416 \leq \mu_{ob} \leq 1,714.17$ $36.25 \leq P \leq 12,750.02$ $36.25 \leq P_b \leq 7,303.01$

- when $\mu_{ob} \leq 50$

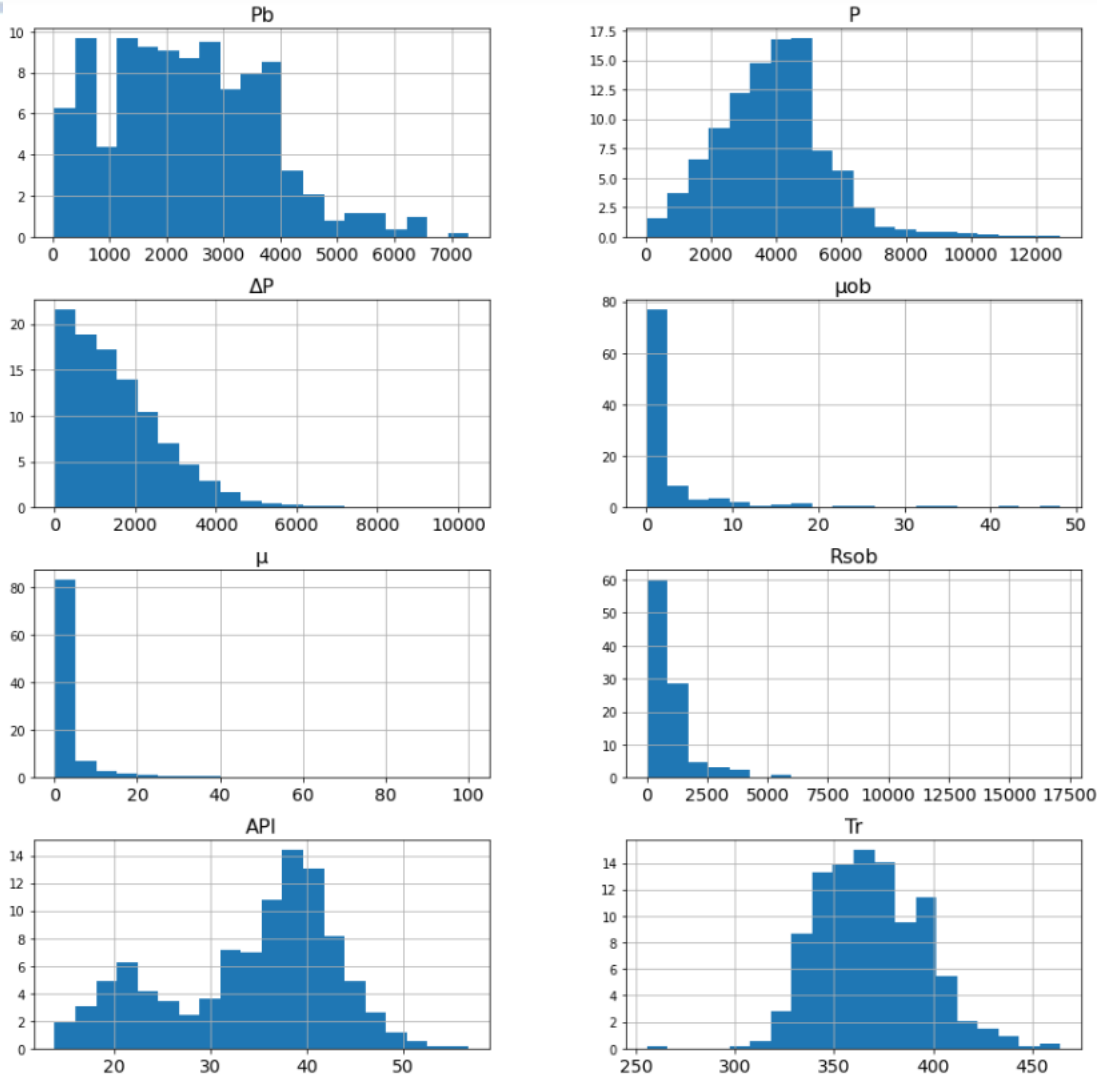


Figure 3: $0 \leq GOR \leq 17,118.6$ $13.82 \leq \gamma_{API} \leq 56.84$ $255.37 \leq T \leq 464.25$ $0.0416 \leq \mu_o \leq 100.1$ $0.0416 \leq \mu_{ob} \leq 48.15$ $36.25 \leq P \leq 12,750.02$ $36.25 \leq P_b \leq 7,303.01$

3.2 Literature evaluation on our dataset

In this section all literature correlation equations will be evaluated when applied to our dataset. As mentioned before, the evaluation will be done both on the whole dataset and on the subset consisting only of bubble point viscosities less than 50 cp. The evaluation will be run using statistical error analysis function. The graphical error analysis and more specifically predicted versus measured values of viscosity, as well as average absolute error vs Pressure differential will be presented in the APPENDIX for each and every literature correlation.

3.2.1 Statistical and Graphical Error Analysis Methods

Statistical and graphical error analyses are the two sole methods used to evaluate the efficiency of oil viscosity correlation equations. In fact, every paper cited in this work, has had its correlation evaluated with these methods. Statistical error analysis determines the overall accuracy of predicted oil viscosity by using basic statistics. In this work average relative error, ARE (or AE as seen in some tables) and average absolute relative error, AARE (or AAE as seen in some tables) will be used. They were considered to be sufficient since ARE is a measure of systematic error or bias and AARE is a measure of random error. A high percentage value of ARE indicates that the correlation under evaluation has a tendency to predict values that are either lower or higher than the measured value. A high value of AARE on the other hand indicates that the predicted values are very far apart from the measured ones. Ideally, both of them need to be close to zero and specifically ARE should be way closer to zero than AARE.

$$\text{ARE} = \frac{100}{N}\sum_{i=1}^{N}\frac{\hat{\mu}-\mu}{\mu}$$

$$\text{AARE} = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{\hat{\mu}-\mu}{\mu}\right|$$

where $\hat{\mu}$ is the predicted value of viscosity and μ is the measured value of viscosity.

One method of graphical error analysis is plotting predicted vs experimentally measured oil viscosity. The perfect correlation line is a straight with a 45° angle. The better the correlation, the more the points on the graph lie closer to this line. Another graphical error analysis tool that will be used is plotting ARE against oil features such as pressure differential, bubble point viscosity, GOR and API gravity. All of them are available in the APPENDIX A at the end of this thesis.

3.2.2 Correlations with bubble point viscosity and pressure

Due to the increased number of correlations studied, they will be divided into two categories. The first includes those who only utilize bubble point viscosity and pressures as input, and the other category being the ones with correlations using in addition GOR, API gravity and dead oil viscosity, or a combination of them. In the APPENDIX many more graphs will be provided along with statistical error analysis tables but only on points where $\mu_{ob} \leq 50$ because this is the cut-off point where our models will be compared against those of the literature.

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Beal (1946)	-1.00	10.48	4.13	9.68
Kouzel (1965)	1.57	7.75	4.83	6.26
Vazquez and Beggs (1976)	8.12	17.60	10.50	16.29
Khan (1987)	0.81	9.51	5.70	7.66
Petrosky (1990)	-0.00	23.72	9.07	21.91
Kartoatmodjo and Schmidt (1991)	-5.48	8.11	6.61	7.22
Orbey and Sandler (1993)	-3.37	8.69	5.17	7.76
Kouzel Modified (1997)	-1.83	7.09	4.39	5.86
Hossain (2005)	3.86	8.35	6.29	6.71

Figure 4: Statistical error analysis on full dataset

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Beal (1946)	-1.76	5.56	3.51	4.65
Kouzel (1965)	1.80	6.97	4.50	5.63
Vazquez and Beggs (1976)	8.56	17.15	10.28	16.17
Khan (1987)	1.62	7.96	5.11	6.31
Petrosky (1990)	1.16	22.92	8.21	21.43
Kartoatmodjo and Schmidt (1991)	-5.78	7.16	6.39	6.62
Orbey and Sandler (1993)	-2.58	6.92	4.43	5.91
Kouzel Modified (1997)	-1.70	6.32	4.04	5.15
Hossain (2005)	4.27	7.54	6.02	6.24

Figure 5: Statistical error analysis on points with $\mu_{ob} \leq 50$

3.2.3 Correlations with additional input

The same statistical error analysis will now be performed on the correlations that utilize additional PVT properties as input. Caution needs to be taken here, since dead oil viscosity hasn't been measured or provided in our dataset at all. For this reason Glaso's correlation (Glaso, 1980) will be used for the estimation of dead oil viscosity's values. This means that there is some uncertainty about the true performance of correlations that utilize dead oil viscosity as input. Furthermore, reports that have missing values for GOR, API and Temperature will be removed when a correlating equation that uses them has to be applied.

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Labedi Libya (1982)	-8.34	10.07	8.66	9.79
Labedi Nigeria (1982)	-8.93	10.79	9.65	10.15
Al-Khafaji (1987)	-5.26	13.54	8.15	12.03
Abdul-Majeed (1990)	17.71	157.55	24.82	156.59
De Ghetto (1994)	-8.61	17.29	10.91	15.94
Almehaideb (1997)	2.09	12.55	7.17	10.51
Elsharkawy (1999)	-10.08	10.43	10.11	10.39
Dindoruk and Christman (2001)	NaN	NaN	NaN	NaN

Figure 6: Statistical error analysis on full dataset

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Labedi Libya (1982)	-7.51	8.27	7.83	7.96
Labedi Nigeria (1982)	-8.11	9.19	8.85	8.48
Al-Khafaji (1987)	-4.33	12.09	7.30	10.57
Abdul-Majeed (1990)	18.33	159.17	24.05	158.41
De Ghetto (1994)	-5.47	9.78	8.00	7.85
Almehaideb (1997)	2.87	11.49	6.64	9.80
Elsharkawy (1999)	-9.27	8.78	9.31	8.74
Dindoruk and Christman (2001)	5.50	12.96	7.13	12.14

Figure 7: Statistical error analysis on points with $\mu_{ob} \leq 50$

3.2.4 Conclusions

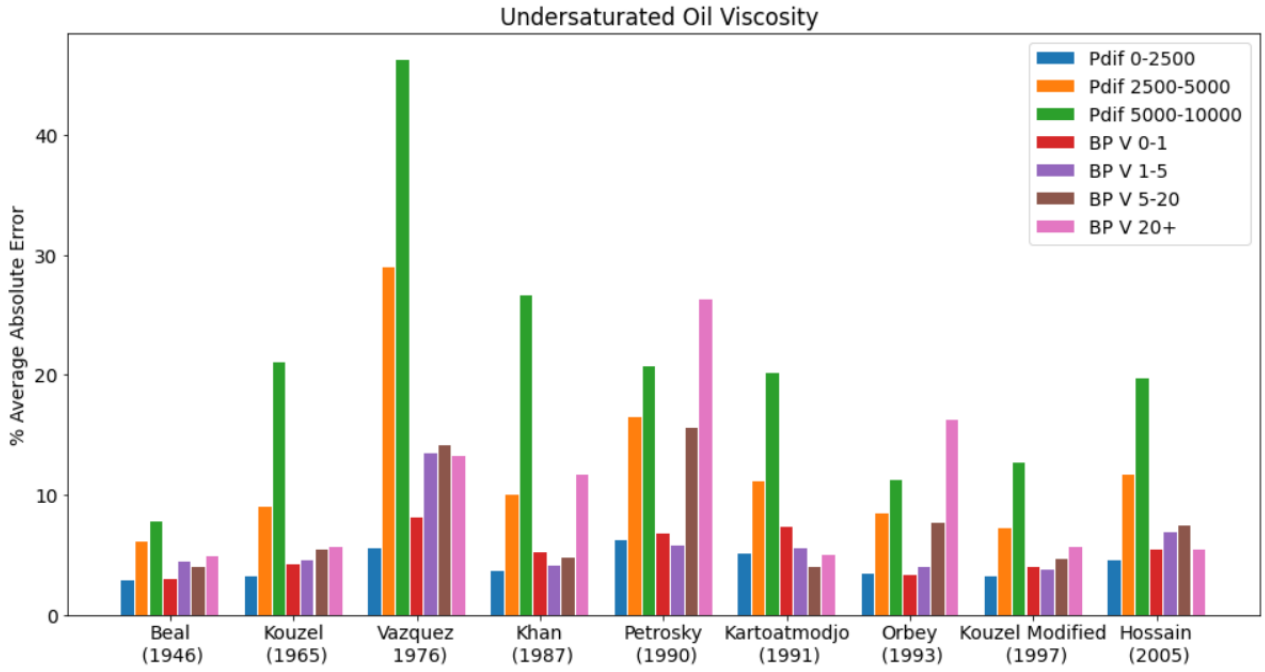


Figure 8: Statistical error analysis by category on points with $\mu_{ob} \leq 50$

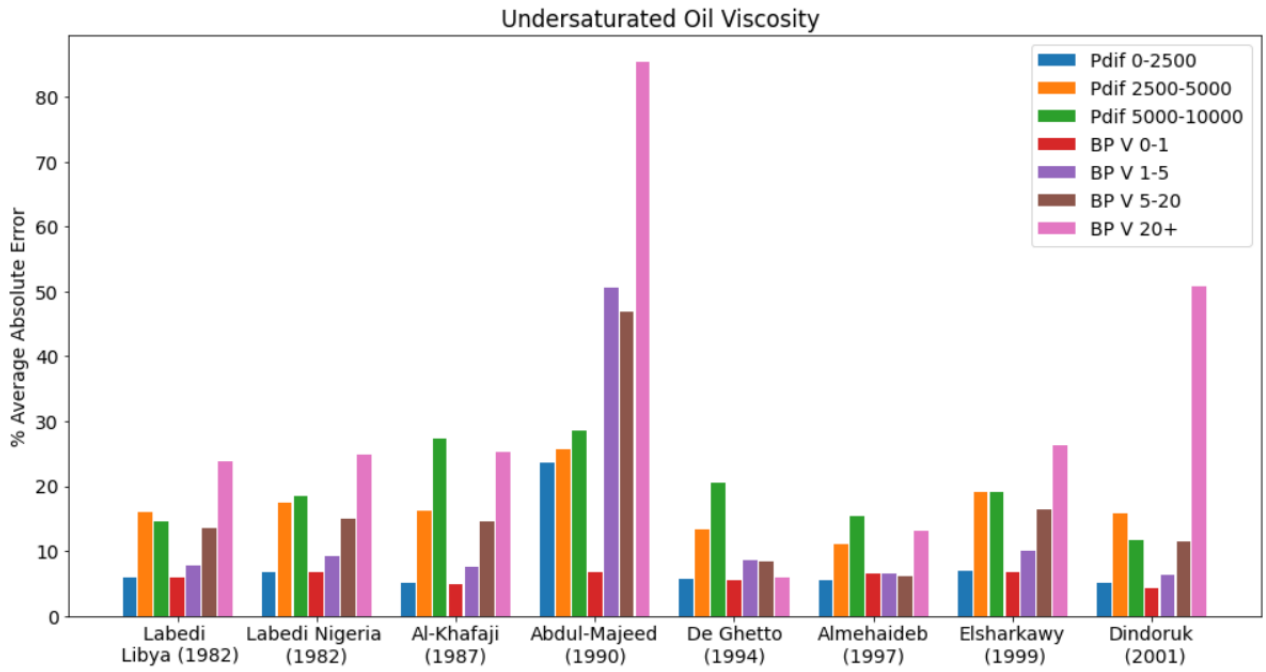


Figure 9: Statistical error analysis by category on points with $\mu_{ob} \leq 50$

where Pdif stands for Pressure differential (P-P_b) and BP V stands for bubble point viscosity. The values next to them indicate the variable's range in which the AAE

was calculated. The colored bars show the mean value of the variable's error inside this range.

In this chapter, the efficiency of the existing correlations on our dataset was briefly examined. More graphs are available for a more detailed examination in the appendices A and B. The following can be concluded:

1. It is clear, yet somewhat surprising that the simpler correlations, that don't utilize additional inputs, have better overall performance. Especially Beal's and (Modified) Kouzel's correlations, which are some of the oldest ones in the literature perform the best.

2. From the eye test it seems that lower values of AARE are exhibited from correlations which utilize pressure differential rather than pressure ratio.

3. As Pressure difference and bubble point viscosity is increased, the errors are increased as well, at almost every correlation.

4. Extremely high values of error are produced by few value ranges in some correlations. Abdul-Majeed's correlation error is very high when GOR and Bubble point pressure's values are close to zero. The same can be said for Dindoruk and Christman 's correlation regarding those values, albeit to a much lesser extent. Furthermore the latter's correlation generalizes very poorly to high values of bubble point viscosity, with the ARE reaching up to 25!! orders of magnitude more than it should. It is somewhat understandable since it was produced for way lower values of bubble point viscosity, but it still remains remarkable. Furthermore,a very high error for oils with high GOR is produced from Almehaideb's equation.

5. For other notable patterns, some form of systematic error within the analysis needs to be looked into. Viscosity as pressure differential increases is overestimated by Vazquez and Beggs, Abdul-Majeed, Almehaideb and Dindoruk. On the other hand,the opposite happens with Kartoatmodjo and Schmidt, Obey and Sandler, Labedi, Al-Khajafi and Elsharkawy , and the viscosity's values are constantly underestimated. Most of those that underestimate it, tend to give better values as the oil becomes lighter. The use of dead oil viscosity from a predictive model needs to be considered here, which plays a part in painting a worse picture for the correlations that use it (Labedi,De Ghetto, Elsharkawy and Alikhan). However, the way that those 3 correlations provide similar forms of systematic error, makes us think that this error may be attributed to the way dead oil viscosity was calculated. ARE seems to be decreasing the higher the bubble point pressure is. This can be seen in the ARE plot of Vazquez and Beggs, Dindoruk, Kouzel and Hossain.

6. Finally, regarding the graphical error analysis, when plotting calculated vs experimentally measured oil viscosity, where the ideal expectation was for lines that go straight with an inclination of 45°, there are a few important things to be said. As expected the correlations with the best AARE seem to be closer to the ideal. Straight lines are not exactly provided by other

correlations, but instead upwards or downwards curves are seen. Finally some correlations resemble a collection of straight lines. When this occurs, it can be concluded that the value of the undersaturated oil viscosity is increased only marginally by the correlation compared to what is needed as pressure changes.

4 Machine Learning methods

In this chapter the machine learning methods that were used in order to create our very own regression models will be presented. These methods were developed on our dataset. Some of the most popular machine learning libraries such as scikit-learn, TensorFlow Keras and others were used to build our models. These models will not be presented in terms of an explicit functional form like the ones from the literature, due to their complexity. However, enough information will be given on their structure, training and implementation.

Following the literature's correlations, only the bubble point viscosity and pressure differential will be used as inputs to our models at first. This will give us the chance to have our models fairly compared to those from the literature using these exact inputs. Extra inputs will then be added to our models to examine if they actually perform better on this dataset. Dead oil viscosity is not given on our list of data so considering Temperature is predominantly used as an input in dead oil viscosity correlations, reservoir Temperature will be used as input in our models. We need to be reminded that generally the literature models' performance was better on this dataset when they were simpler in form, i.e. bubble point viscosity and pressure differential or ratio were only used as inputs.

4.1 Feature Engineering

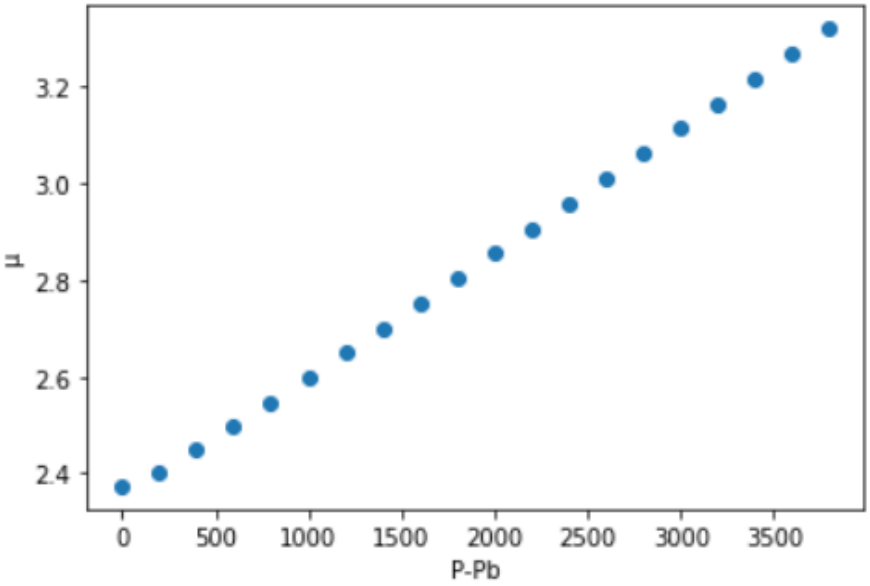
By far the hardest part of working on this project was the feature selection and engineering. While working several sources of possible errors were identified, that if left untreated would cause our models to either be unable to generalize to new datapoints or just be outright bad regression models. Furthermore, features needed to be transformed in a way that would grant the best overall results among most regression models fitted.

4.1.1 Train-Test split

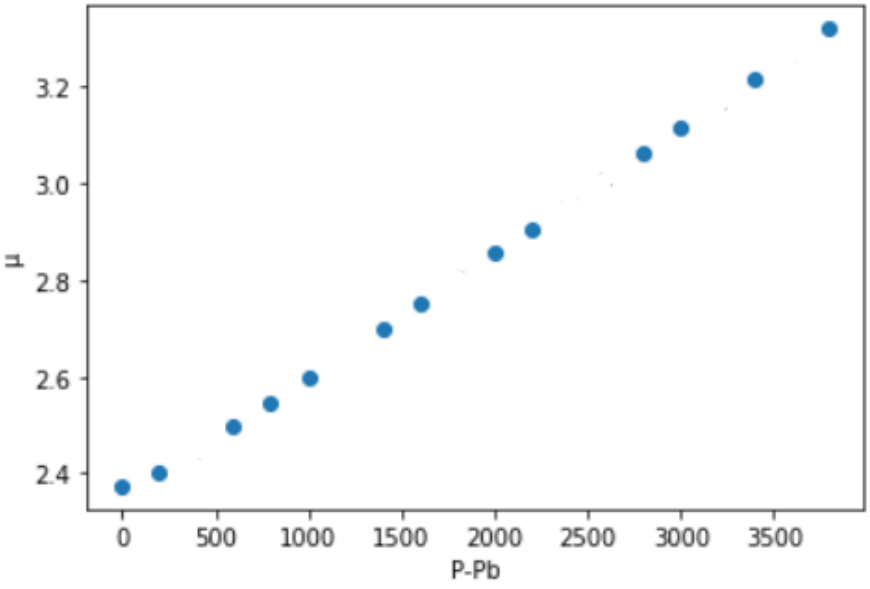
To avoid overfitting, when machine learning methods are utilized, it is common practice for its performance to be judged on a new set. Furthermore many machine learning algorithms get fitted on an original set and tuned on another set to make them able to generalize better. This means that our original dataset needs to be split into three sets. The one against which fitting happens is called the Train set, the one where the tuning happens is called Validation set and finally the one against which the model is evaluated is called the Test set. The dataset was split following some of the most commonly used rules. 20% of the original set was kept as a test set, 16% as the validation set and 64 % as the training set.

At first, the AARE of our models exhibited such an abnormally low value, that would be deemed alarming by even the most optimistic person. The issue was that when splitting thousands of data points, it needs to be ascertained that every single point that belongs to a certain fluid, must be part of a set. For example, 20 points from a fluid report can't be split between the sets because that defeats the purpose

of this work. The goal is to find out how undersaturated viscosity will change starting from the bubble point and up to a maximum pressure (presumably the reservoir pressure). The problem can be summed up to the images below.



(a) all pressure data points of a single fluid



(b) selected pressure data points of a single fluid in train set

Figure 10: The result of incorrect data splitting

It's very easy to see that the missing data points from the figure on the bottom could be effortlessly found by even a not so efficient machine learning algorithm. This small issue, although easy to find out, goes on to show the caution needed when dealing with Machine Learning and not getting excited without seeking out

all possible missteps made along the way.

The process of splitting is not done yet. It is already mentioned that the majority of our points (more than 85%) have a bubble point pressure viscosity of less than 5 cp. Splitting is done randomly, so we could fall into the trap of getting one of our smaller sets to have an unrepresentative amount of fluids with bubble point pressure viscosity of more than 5 cp. This means that our test or validation sets could potentially have few, or even no fluids across a big range of viscosities where the oil is more viscous and its viscosity changes with pressure in an abnormal way. In order to avoid this a technique called stratified split was used, where the sets were specifically split into groups having the same percentage of points with viscosities over 5 cp.

4.1.2 Missing Values

Another important issue that arose during this work was the treatment of missing features' values. Specifically, around 40 fluid reports provided no value for either the GOR or the API gravity or both. In the previous chapter those reports were simply dumped (where those properties were not utilized), because what was needed was the evaulation of the performance of literature correlations. Now that new models of our own need to be developed, dumping around seven percent of the data points might not be the most optimal idea, especially since those features don't seem to be the most important ones. The importance of each feature can be measured in a plethora of ways, however in this work Pearson's correlation coefficient was used, since the need to transition into another and possibly better one didn't arise. Pearson's correlation coefficient will be presented in subsection (4.1.3).

Back to the missing values issue. We identified 3 possible strategies to overcome the problem of missing values.

- Remove all data points that have missing values. As said before, this option is the most conservative and technically the most correct, but removing a big part of our dataset would not work in our favor. Indeed this method produced worse results when tried.

- Resort to a simple imputing strategy. This usually works quite well in machine learning models. If a value is missing, then a value that is representative of the population we can just be introduced in its place, such as the average, the median, etc. But we felt like there was a better treatment available at our disposal.

- Use simple machine learning tools to impute values. In our case a Random forest regressor was used to impute the missing values. There is no metric to judge whether the model worked well or not but the values predicted from the models were well within the range of the feature. This method produced the best results overall, hence we decided to stick with it.

The imputed values and the original ones will now be plotted against bubble point viscosity to see if the imputed values follow the trend or distribution of the population.

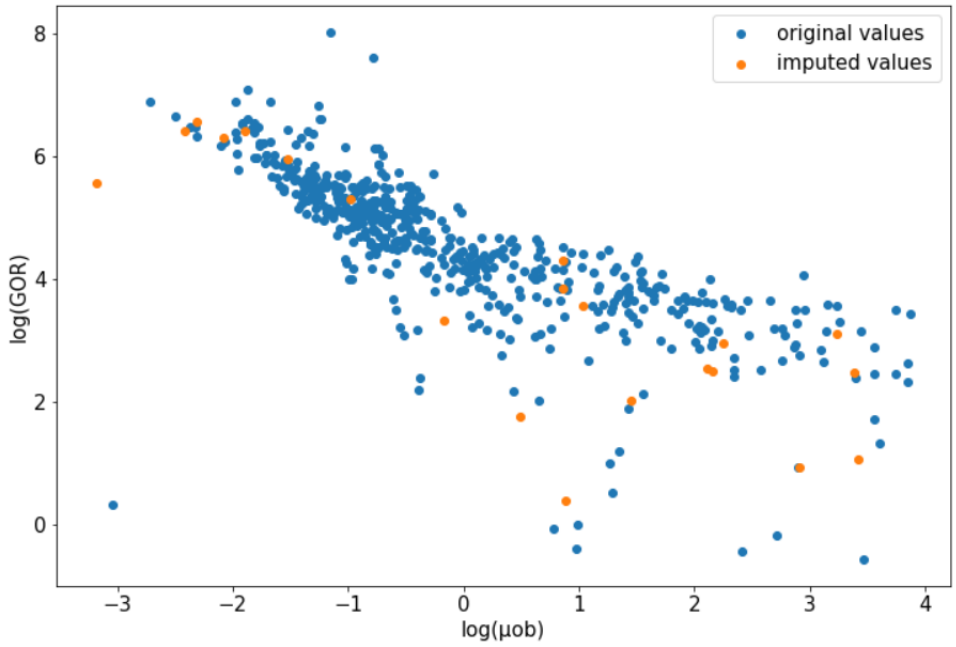


Figure 11: Imputed values of GOR

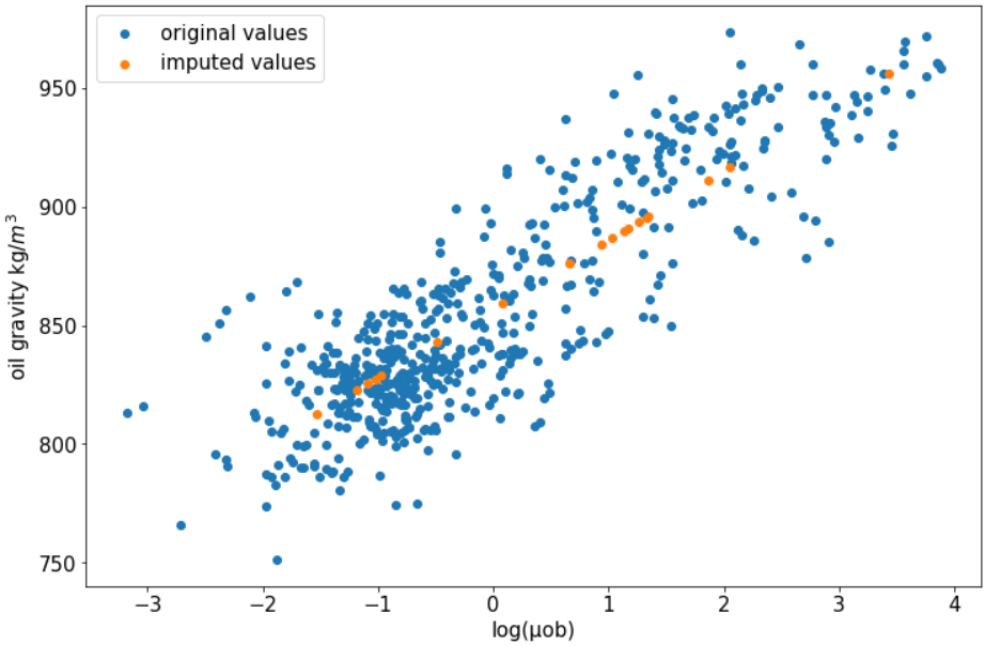


Figure 12: Imputed values of oil gravity

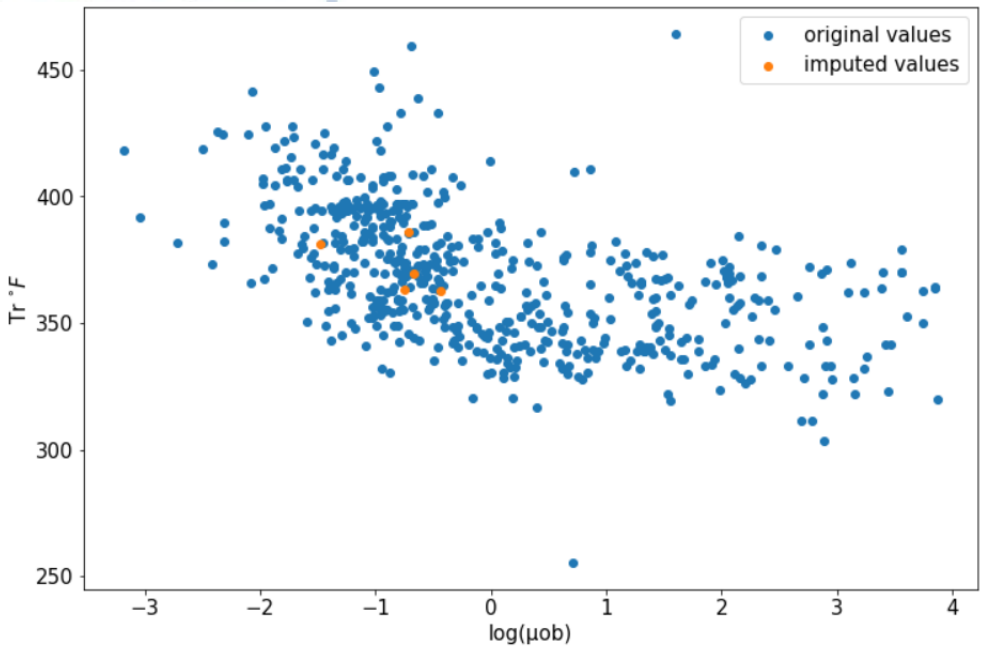


Figure 13: Imputed values of temperature

The imputed values fit seamlessly with the original values when plotted against bubble point viscosity which gives us confidence about our strategy regarding the imputing of missing values.

4.1.3 Feature optimization

Now is the time to tackle the hardest part of feature engineering. The optimization of our features and targets before introducing them to the machines in order to produce the best possible result. Our data need to be transformed in a way that makes them optimally correlated to the target, which is of course the undersaturated oil viscosity. This is very important because the higher the correlation, the better the efficiency of the models.

It has already been mentioned that the metric in which the features will be judged is Pearson's correlation coefficient. This is a measure of linear correlation between two sets of data. It is defined by the ratio between the covariance of two variables and the product of their standard deviations. Essentially, it is a normalized measure of the covariance, as its value ranges from -1 to 1. Mathematically it can be expressed with the following formula

$$\rho_{X,Y} = \frac{\mathbb{E}[(X - \mu_X)(Y - \mu_Y)]}{\sigma_X \sigma_Y}$$

where

- $\rho_{X,Y}$ is the Pearson's correlation coefficient
- $\mathbb{E}$ is the expected value

- μ is the mean of the variable that is underscored
- σ is the standard deviation of the variable that is underscored

At first our original data exhibit the following correlation coefficient with undersaturated oil viscosity.

μ	1.000
μob	0.976
ΔP	0.168
P	-0.205
Rsob	-0.241
Tr	-0.246
Pb	-0.388
API	-0.594

Figure 14: correlation coefficient table to viscosity

A very high correlation with our target and bubble point viscosity is observed as expected. However pressure difference has the lowest correlation of them all. This is bad news since pressure difference (and Pressure) is the only feature that changes as viscosity changes for a certain fluid. Furthermore API gravity has a strong negative correlation with viscosity. We need to elaborate further to see why this happens.

Firstly, we will look into the scatter plots of the features and the viscosity.

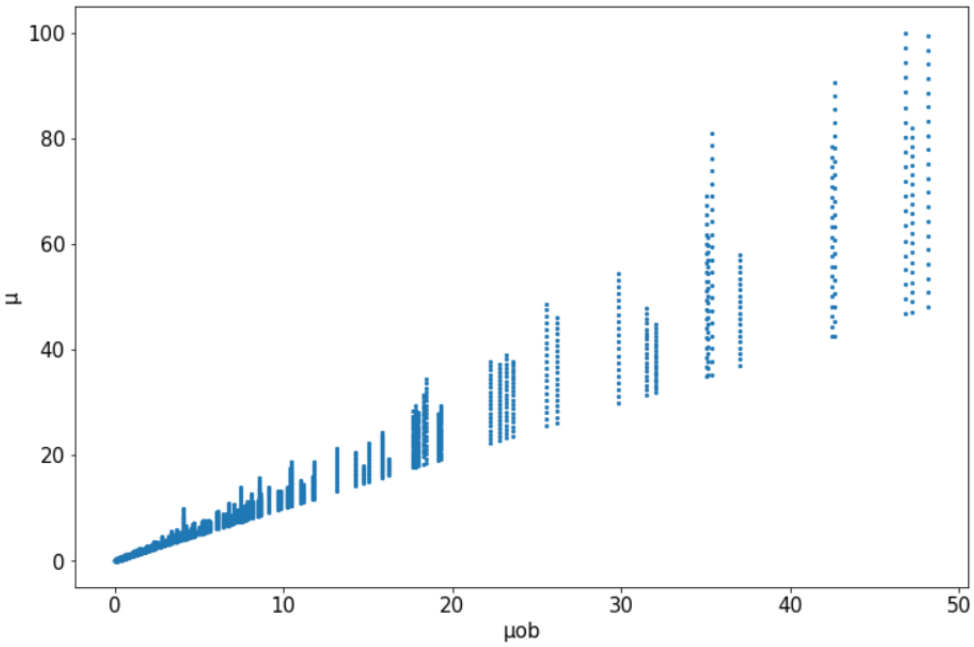


Figure 15: scatter plot: viscosity to bubble point viscosity

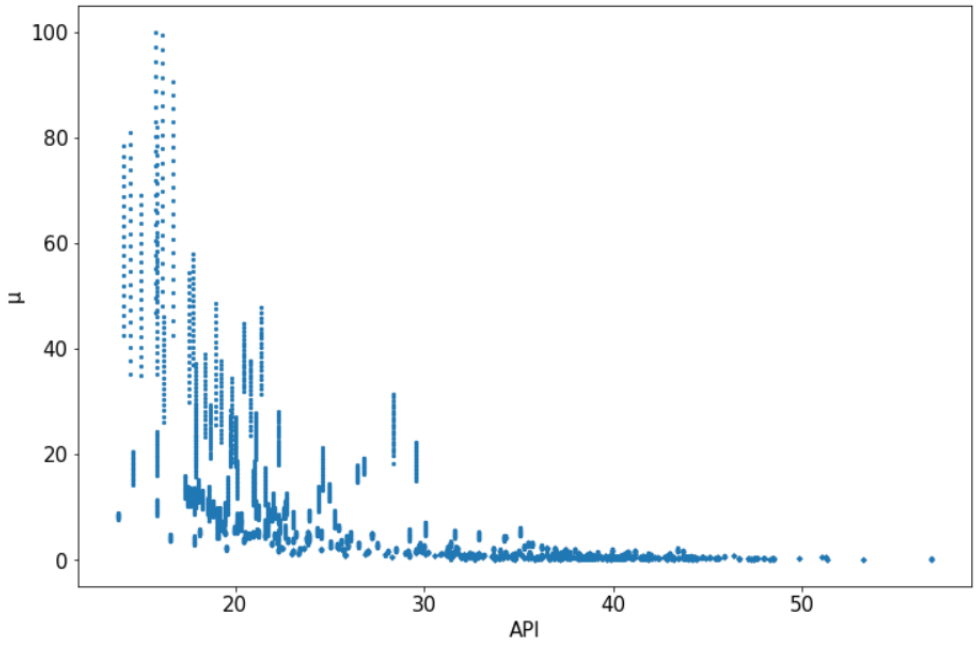


Figure 16: scatter plot: viscosity to API

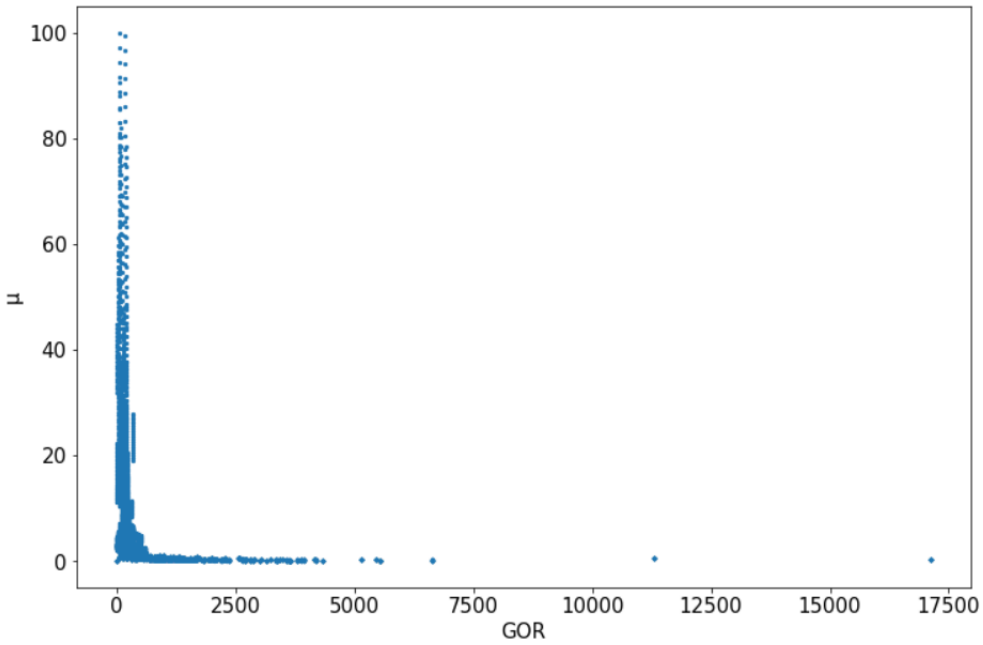


Figure 17: scatter plot: viscosity to GOR

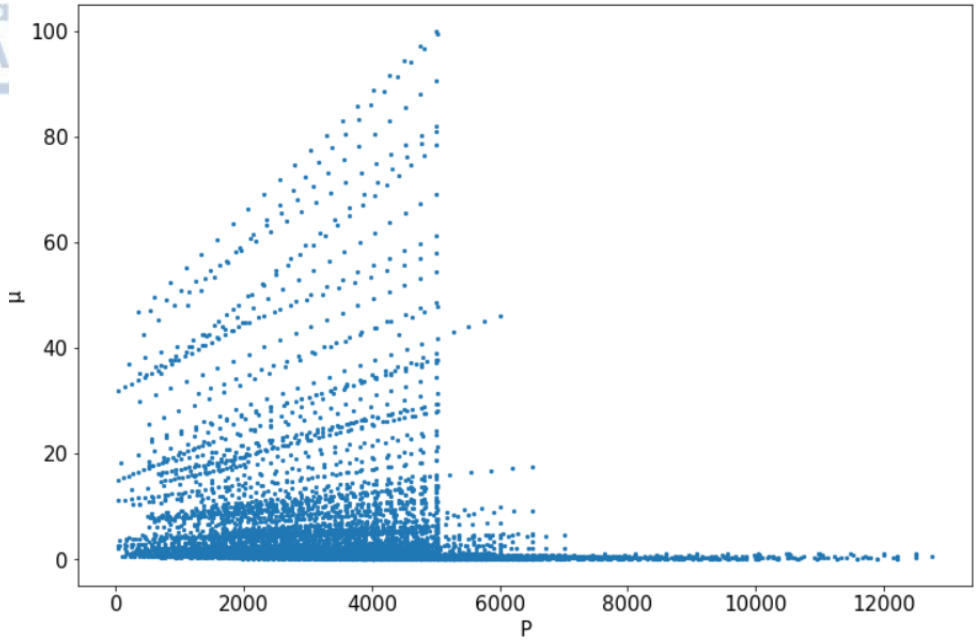


Figure 18: scatter plot: viscosity to Pressure

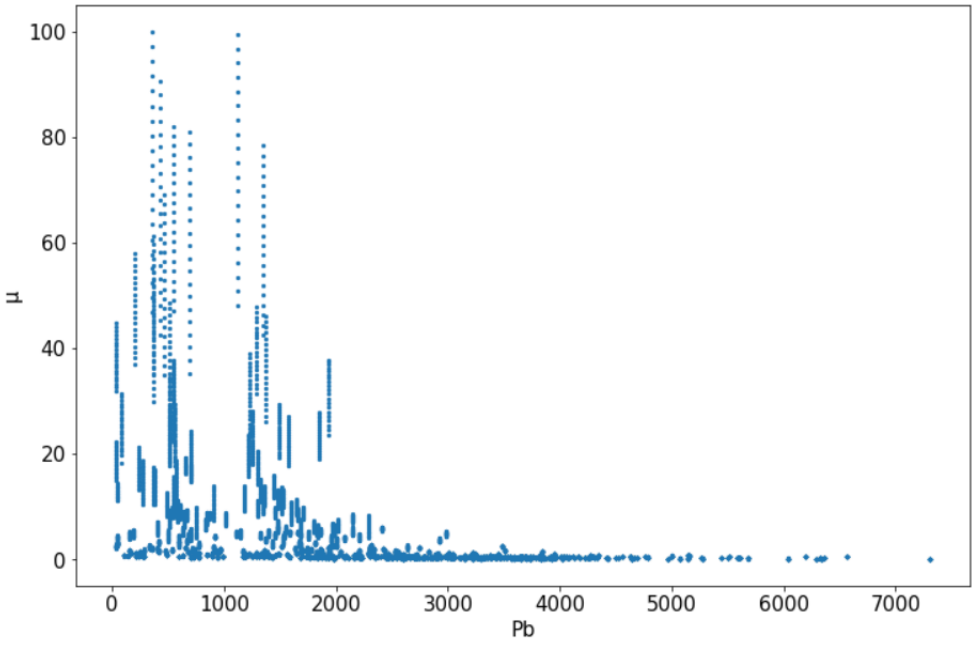


Figure 19: scatter plot: viscosity to bubble point pressure

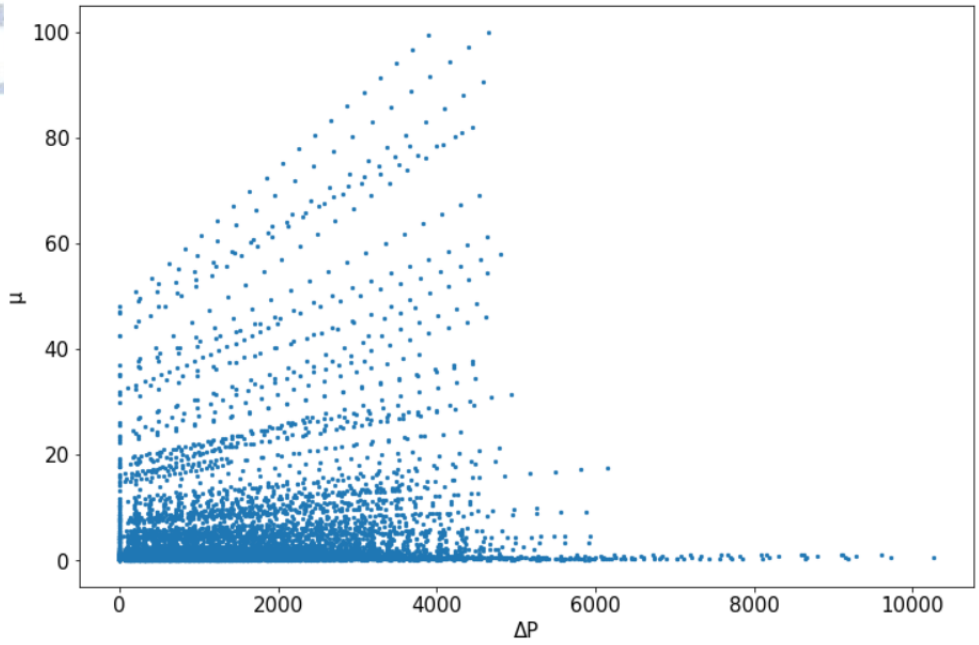


Figure 20: scatter plot: viscosity to pressure difference

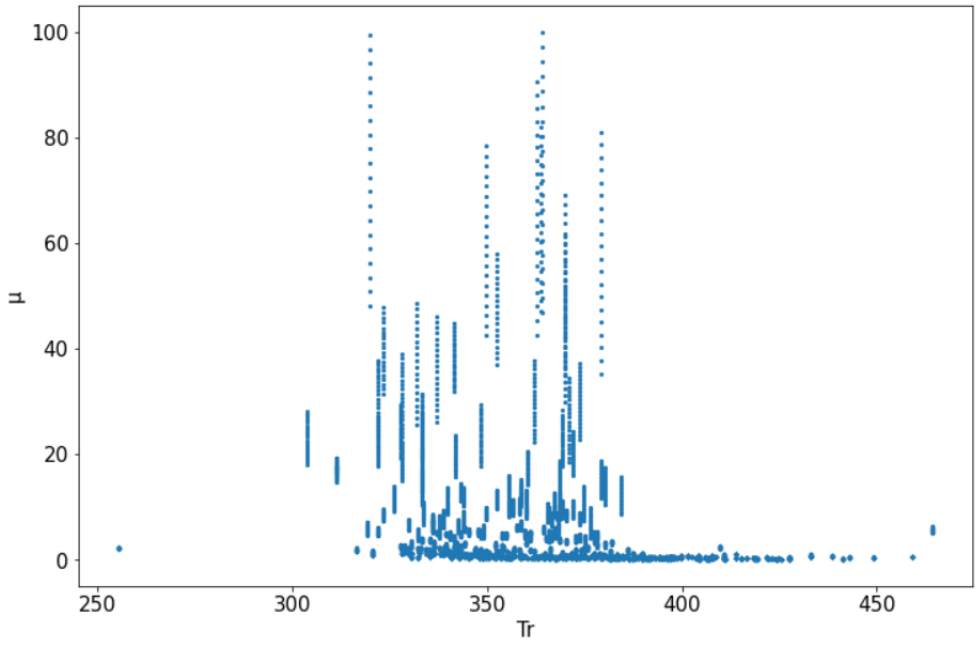


Figure 21: scatter plot: viscosity to temperature

Even though API gravity has a highly negative correlation to viscosity, it doesn't directly impact how viscosity increases from the bubble point up to the maximum pressure. There is however a trend that fluids with a low API (heavy fluids) tend to be more viscous and also that their viscosity increases more rapidly as the pressure increases. Same thing seems to happen for the GOR. While pressure has a low correlation we can see that for every fluid there is an almost linear (slightly curved

at most) relationship with pressure differential and viscosity and furthermore, the higher the viscosity the bigger its gradient with pressure difference gets.

While these trends were good enough to give us some respectable models it seemed as though a better optimization could be done. Another reason to feel this way was the correlation matrix of bubble point viscosity and the other features.

μob	1.000
μ	0.976
ΔP	0.110
Rsob	-0.264
P	-0.272
Tr	-0.277
Pb	-0.417
API	-0.636

Figure 22: correlation coefficient table to μ_{ob}

It seems as though most features are better correlated to bubble point viscosity than viscosity in general. Another issue that was expressed in many models was that the viscosities that they predicted had high AARE at pressures close to the bubble point. It needs to be reminded that bubble point viscosity is assumed to be known, so a wrong prediction close to it is a sign of the model or the data optimization being not optimal. A suspected reason for this failure is the vast variation in viscosity. The models need to predict viscosity values varying from 0.1 cp up to 100 cp. This is a variation of 3 orders of magnitude.

This issue was worked around by changing the target. Instead of trying to predict the viscosity, the viscosity over bubble point viscosity was instead the next target being predicted, a form of normalized viscosity $\left(\frac{\mu}{\mu_{ob}}\right)$. This greatly altered our data's correlation coefficients and it uncovered the correlation of our target and pressure difference. It needs to be kept in mind that many of the literature's regression methods are of the form $\mu \propto \mu_{ob} \cdot f(\Delta P)$ or $\mu \propto \mu_{ob} + f(\Delta P)$ so this way of working is supported by the existing methods as well. Additionally, the normalization affects the correlation coefficients which now express the linearity of μ with ΔP provided that μob is known, that is by incorporating the dependent rather than the direct probability.

μ	1.000
ΔP	0.752
μob	0.398
P	0.331
Tr	-0.059
Rsob	-0.135
Pb	-0.259
API	-0.294

Figure 23: correlation coefficient table to μ/μ_{ob}

One would think that in the current form correlations are actually worse than before. However scale is important here. Pearson's correlation coefficient $\rho_{X,Y}$

depends on the X,Y scale. Before this change, correlations were better, yet the target that was predicted varied in a range of three orders of magnitude. Now, data's correlation to the target is worse, but the target's values range from 1 to 2.5. There was a huge improvement with every model fitted, so it was a no brainer to continue with this change. The only issue that remains now, is that for a certain fluid there is still no direct impact between its viscosity and variables such as Pb,T,API, GOR. Even if some trends are seen just like before in the scatter plots, our models could hurt from it. Those features needed to be transformed in a way so they would change with viscosity. The simplest solution thought of was multiplying every feature with pressure difference. The features were then optimized by using a function of them as opposed to their original values. So for example instead of creating the feature $\Delta P \cdot GOR$, $\Delta P \cdot \ln(GOR)$ was created. It was then just a matter of finding the best function of every feature for the best correlation coefficient to be produced.

μ	1.000
μob	0.894
API	0.891
Rsob	0.887
Pb	0.882
Tr	0.879
P	0.878
ΔP	0.868

Figure 24: correlation coefficient table to μ/μ_{ob} with modified features and target

This is the final correlation coefficient table that was created. A good indicator that our models are going to perform well is that in this feature shape, almost all features have a good normal like distribution, if the spike at point 0 is to be excluded which happens due to the bubble point data. Machine learning models tend to work better on data that are normally distributed. This last step concludes our feature selection and optimization procedure. It should be mentioned the exact procedure followed was not as straightforward as it appears in this chapter. It took many steps back and forth to judge our models, optimize them, optimize the features, repeat for the optimal result along most regression models.

This last step can be thought of as the data scientist's approach. It is a vastly different approach to the reservoir engineer's one that was used before, when changing our target. The reservoir's engineer approach was essential in transforming the problem in a physically sound way in order to uncover what can be seen as the true relation that governs the task in question. When introduced, it instantly changed the potential of machine learning algorithms's usage in this work from barely adequate to the go to method. Before tackling the problem as a reservoir engineering one, only using the data scientist's approach, the results were limited partly due to the small amount of data available at our disposal and partly due to not transforming them in thinking of the problem as engineers. Just to be clear, the data scientist's approach should not be underestimated. Given enough time, effort and data, the results could have been similar even when used on its own. It is however undeniable, that the most important factor in succeeding in this task was taking a step back and thinking around the issues in a different manner than a pure data

scientist would. That is, the ability for engineering and physics driven concepts to be combined with data driven, machine learning approaches.

4.2 Linear models

Speaking of the data scientist's approach, now comes the part where our models will be introduced by category. They will then be evaluated on the test set and finally, the top performers will be selected and judged against the top literature performers to see if our models perform better on our test set. It needs to be reminded that our models will be evaluated on a set that they have not been trained before so they should in theory have no advantage against the correlations from the literature. Furthermore any AARE presented in this chapter should not provide an exact proof of the superiority or the inferiority of our models, since they are evaluated on a subset of the original set. It will just give us a fair understanding of which regression model performs best so that it will be chosen and then evaluated against the literature's correlations in the next chapter.

Linear models are used when a linear relationship between the output and the features is targeted to be found. They usually perform worse than other more powerful regression models, but they are still included due to their popularity and ability to be tweaked into obtaining non linear better models.

4.2.1 Linear Regression

Having cleared this up, the simplest linear regression method will be presented. In general, linear regression is a method that fits a linear model to minimize the residual sum of squares between the observed targets in the dataset and the targets predicted by the linear model. Since multiple input variables are available, our models can be characterized as multiple linear regression models. The model will now be described below.

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

where $\hat{y}$ is the predicted value, n is the number of features, θ_i are the coefficients and x_i are the feature values. This equation can be written in a compressed vector formula:

$$\hat{y} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \boldsymbol{\theta}^{\intercal} \cdot \mathbf{x}$$

To optimize the model the values of θ_i that minimize the MSE or the RMSE of the model need to be obtained. The MSE of a linear regression hypothesis $h_{\boldsymbol{\theta}}$ on a training set $\mathbf{X}$ is

$$MSE(\mathbf{X}, h_{\boldsymbol{\theta}}) = \frac{1}{m} \sum_{i=1}^{m} \left(y^{(i)} - \boldsymbol{\theta}^{\intercal} \mathbf{x}^{(i)} \right)^2$$

where X is the matrix of all feature values of all instances in the training set. Every row contains an instance transposed, meaning that the i^{th} row contains the transpose of $\mathbf{x}^{(i)}$. In short it looks like this:

$$\mathbf{X} = \begin{pmatrix} \left(\mathbf{x}^{(1)}\right)^\intercal \\ \left(\mathbf{x}^{(2)}\right)^\intercal \\ \vdots \\ \left(\mathbf{x}^{(L)}\right)^\intercal \end{pmatrix}$$

where L is the number of training pairs. To find the value of $\boldsymbol{\theta}$ that minimizes the MSE (cost function), there exists a closed form solution called the normal equation

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^\intercal \mathbf{X})^{-1} \mathbf{X}^\intercal \mathbf{y}$$

where $\hat{\boldsymbol{\theta}}$ contains the values that minimize the MSE, $\mathbf{y}$ is the vector containing all of the targets and $(\mathbf{X}^\intercal \mathbf{X})^{-1}\mathbf{X}^\intercal$ is the Moore-Penrose inverse matrix.

4.2.2 Gradient Descent

Gradient descent is an optimization method that firstly measures at the current estimate the local gradient of the cost function with respect to the parameter vector and secondly makes a step towards the direction where the gradient descents. When the gradient goes to zero, a minimum has been found. It starts by a random initialization of $\boldsymbol{\theta}$ and improves on it gradually, atempting to decrease the cost function, until the algorithm converges. Learning rate is a hyperparameter of the algorithm and it actually determines the size of each step. If the value is too low, then the algorithm might take many iterations to converge. If it is too high, then it might surpass the minimum, making the algorithm diverge. To implement gradient descent, the partial derivative of the cost funtion with respect to each parameter θ_i needs to be computed. Just to be clear, gradient descent isn't of any use in linear regression since a closed form solution already exists. However, it will be used later on for manifestations where the objective function is not linear any more.

$$\frac{\partial}{\partial \theta_i} MSE(\boldsymbol{\theta}) = \frac{2}{m} \sum_{j=1}^{m} \left(\boldsymbol{\theta}^\intercal \mathbf{x}^{(j)} i y^{(j)}\right) x_i^{(j)}$$

This can be written in vector form as

$$\nabla_{\boldsymbol{\theta}} MSE(\boldsymbol{\theta}) = \frac{2}{m} \mathbf{X}^\intercal (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$$

Once the gradient vector, that points uphill, is calculated, the step is taken to the opposite direction.

$$\boldsymbol{\theta}' = \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} MSE(\boldsymbol{\theta})$$

where η is the learning rate and $\boldsymbol{\theta}'$ contains the updated value of the coefficients. This method has a problem however. In order to compute the gradient at every step, the whole training set needs to be used. This makes the algorithm very slow. So instead of that, stochastic gradient descent (SGD) can be used. This method picks at every step, a random instance in the set and then measures the gradient using this instance only. This algorithm is faster, but its problem is that it will not reach the minimum of the cost function smoothly. It will bounce up and down,

decreasing only on average and once it finds the minimum, it will not stay there, it will bounce around it but stay close to it as well. In our implementation of it no penalty function was chosen, since its use would perform Ridge or Lasso regression which wasn't our objective at this point of the research.

4.2.3 Polynomial Regression

In this model the total number of our features was expanded by using products of two of our features to create new ones. A polynomial model of higher degree could be tried, but their worse performance, plus the exponential increase in features and the risk of overfitting were reasons to deter the work from going to this direction. A way of reducing overfitting even in a second degree polynomial model is its regularization, meaning to reduce its degrees of freedom. Some polynomial models with regularization created will now be looked at.

4.2.4 Ridge Regression

Ridge regression is a constrained version of linear regression. The regularization term $\alpha \sum_{i=1}^{n} \theta_i^2$ is added to the cost function. This is called $\ell 2$ regularization because the regularization term is actually the $\ell 2$ norm of the weight vector $\boldsymbol{\theta}$. The regularization term α that was chosen in our model was 0.2, which indicates our belief that not much regularization was needed in our polynomial model. The cost function is

$$J(\boldsymbol{\theta}) = MSE(\boldsymbol{\theta}) + 0.2\frac{1}{2}\sum_{i=1}^{n} \theta_i^2$$

Just like linear regression, Ridge regression can be performed either by using the closed form equation or by SGD. The closed form was chosen since the quantity of our data is not that big to make the process slow. The closed form solution is as follows:

$$\hat{\boldsymbol{\theta}} = (\mathbf{X}^\intercal \mathbf{X} + 0.2\mathbf{A})^{-1}\mathbf{X}^\intercal \mathbf{y}$$

with $\mathbf{A}$ being the $(n+1) \times (n+1)$ identity matrix, except with a zero in the top left cell corresponding to the bias term.

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

4.2.5 Lasso Regression

Just like Ridge, this method adds a regularization term, this time by using the $\ell 1$ norm.

$$J(\boldsymbol{\theta}) = MSE(\boldsymbol{\theta}) + \alpha \sum_{i=1}^{n} |\theta_i|$$

Lasso doesn't allow for a closed form solution so instead it uses stochastic gradient descent. The cost function is still not differentiable at $\theta_i = 0$, but the following subgradient vector $\mathbf{g}$ is used when this happens.

$$\mathbf{g}(\boldsymbol{\theta}, J) = \nabla_{\boldsymbol{\theta}} MSE(\boldsymbol{\theta}) + \alpha \cdot \text{sign}(\boldsymbol{\theta})$$

$\alpha = 0.001$ was used in our Lasso model.

4.2.6 Elastic Net

Elastic net is a method that combines both regularization parameters and the mix ratio can be controlled. When set to zero, Elastic net is equivalent to Ridge and when set to one, it is equivalent to Lasso.

$$J(\boldsymbol{\theta}) = MSE(\boldsymbol{\theta}) + r\alpha \sum_{i=1}^{n} |\theta_i| + \frac{1-r}{2} \alpha \sum_{i=1}^{n} \theta_i^2$$

In our model Lasso's regularization parameter of 0.001 plus a ratio r value of 0.4 was used. (Géron, 2019)

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Linear Regression	-0.44	5.08	3.29	3.90
Polynomial Regression	0.26	5.68	3.19	4.71
Gradient Descent	-0.64	5.19	3.33	4.04
Ridge Regression	-0.06	4.66	2.92	3.63
Lasso Regression	-0.44	5.15	3.32	3.96
Elastic Net	-0.41	5.04	3.28	3.85

Figure 25: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Linear Regression	-0.43	5.08	3.25	3.93
Polynomial Regression	0.09	5.26	3.21	4.17
Gradient Descent	-0.67	5.32	3.37	4.17
Ridge Regression	-0.12	4.90	3.04	3.85
Lasso Regression	-0.44	5.10	3.27	3.94
Elastic Net	-0.41	5.08	3.26	3.92

Figure 26: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

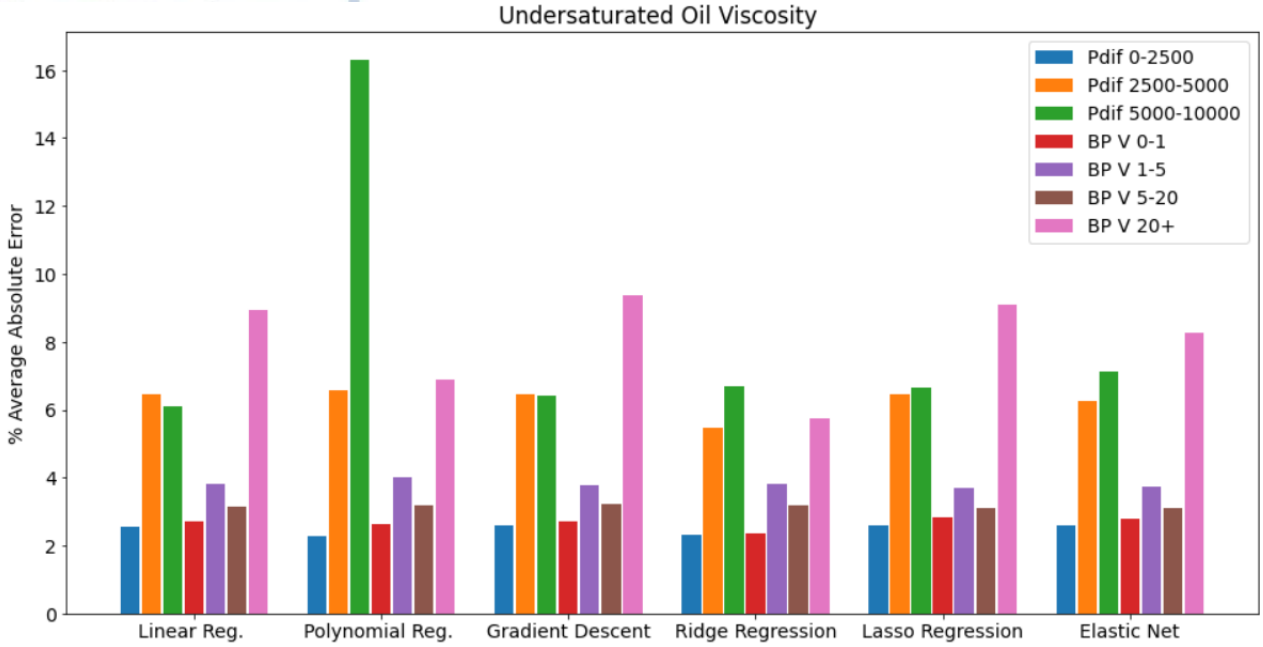


Figure 27: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

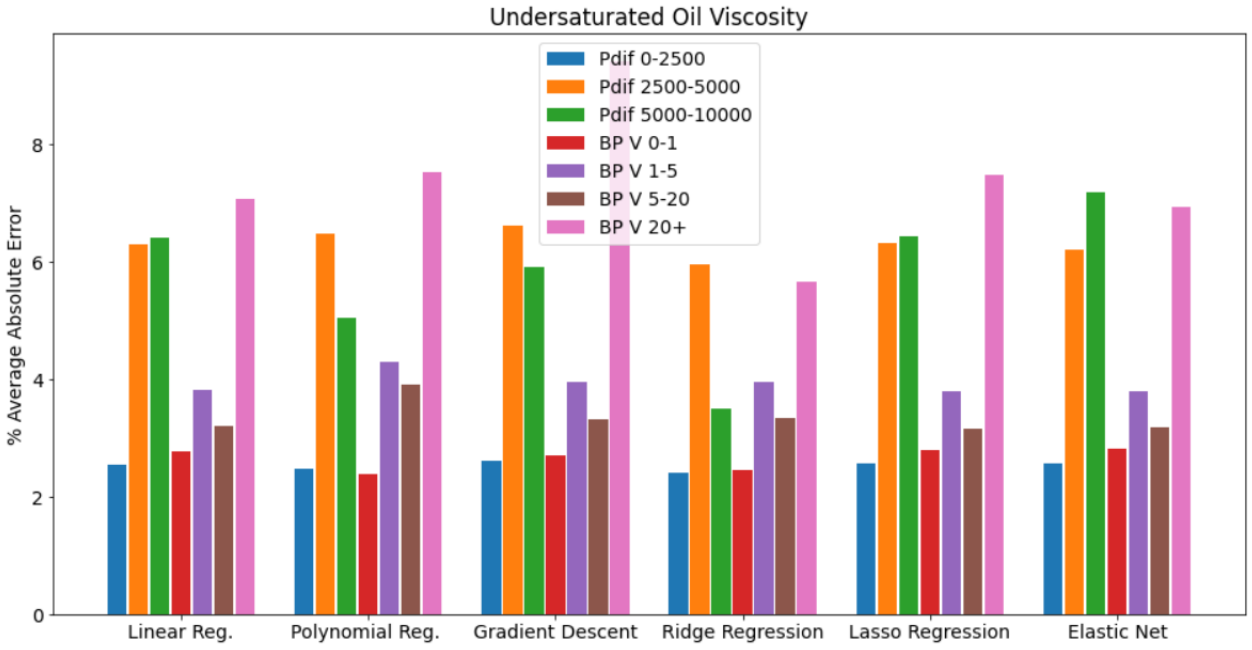


Figure 28: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

A few conclusions can be made here. First of all results are consistently a bit better when using the full set of features. Ridge regression is the top performer among all regression models in almost every value range that the results are shown. There is a big outlier with large pressure differentials' AARE in the full dataset in polynomial regression on the full feature set that drops to a better value when using the reduced feature set. All in all most regression models perform similarly, with

small values of AARE being seen at the smallest bubble point viscosity values and the smallest pressure differentials.

4.3 Support Vector Machines

At first support vector machines were developed for handling classification problems. First of all, it will be explained how they are developed and then their extension into regression. For the explanation below (Fletcher, 2009) will strictly be followed

4.3.1 Theory

Let's say there are L points $(\mathbf{x}_i, y_i)$ for training purposes, with every input $\mathbf{x}_i$ having D features, meaning it is D-dimensional and $y_i = \pm 1$ denoting the class of each point. Now let's assume that the data is linearly separable, meaning that we can draw a (D-1) dimensional hyperplane to separate the data that belong to different classes.

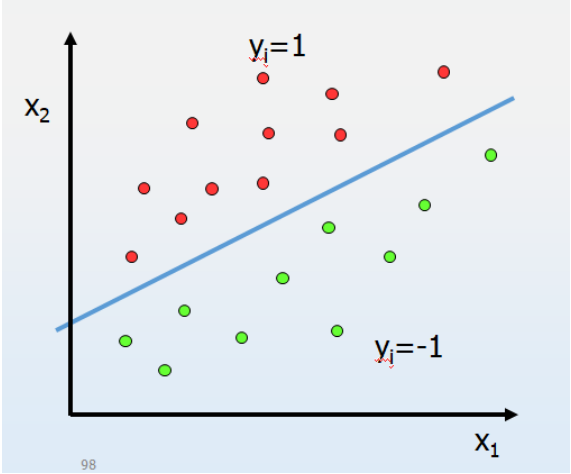


Figure 29: Discriminating Hyperplane

This hyperplane can be described by

$$\mathbf{w}^\intercal \cdot \mathbf{x}_i + b = 0$$

where $\mathbf{w}$ is a vector normal to the hyperplane and $\frac{b}{||\mathbf{w}||}$ is the perpendicular distance from the hyperplane to the origin. Support vectors are the points of each class that are closer to this hyperplane. Implementing an SVM means that b and $\mathbf{w}$ need to be selected in a way that our model predicts a value of $y_i = +1$ beyond a margin on one side of the hyperplane and a value of $y_i = -1$ beyond a margin on the other side of the hyperplane.

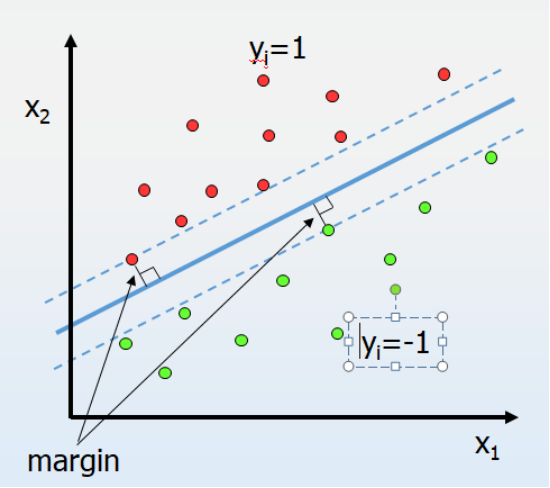


Figure 30: Margin and parallel Hyperplanes

This is because SVM's is developed to predict classes in which new cases are assigned to. The value of y_i indicates the class in which the new instance is a part of. Suppose that this margins are two hyperplanes parallel to the first one. So what is needed is to find the abovementioned variables such that:

$$\mathbf{w}^\intercal \mathbf{x}_i + b \geq +1 \quad \text{for } y_i = +1$$
$$\mathbf{w}^\intercal \mathbf{x}_i + b \leq -1 \quad \text{for } y_i = -1$$

This can be written into a more concise form

$$y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1 \geq 0$$

It follows that the support vectors are the points that lie on those margin hyperplanes, which are called H_1 and H_2 and are parallel to the discriminating hyperplane.

$$\mathbf{w}^\intercal \mathbf{x}_i + b = +1 \quad \text{for } H_1$$
$$\mathbf{w}^\intercal \mathbf{x}_i + b = -1 \quad \text{for } H_2$$

Let d_1 be the distance from H_1 to the discriminating hyperplane and d_2 the distance from H_2 to the discriminating hyperplane. The hyperplane's margin is a quantity equal to the sum of those distances.

$$d_1 = \frac{\mathbf{w}^\intercal \mathbf{x}_i + b}{||\mathbf{w}||} = \frac{1}{||\mathbf{w}||} = \frac{1}{||\mathbf{w}||}$$
$$d_2 = \frac{\mathbf{w}^\intercal \mathbf{x}_i + b}{||\mathbf{w}||} = \frac{-1}{||\mathbf{w}||} = \frac{1}{||\mathbf{w}||}$$

The total margin is equal to $\frac{2}{||\mathbf{w}||}$. The hyperplane needs to oriented to be as far away as possible from the points of each class. In order to do that the margin needs to be maximized. This is equal to minimizing $||\mathbf{w}||$, or minimizing $L = \frac{1}{2}||\mathbf{w}||^2$ on the condition that $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1 \geq 0$

In order to minimize $||\mathbf{w}||$ while respecting the constraints, Lagrange multipliers $\boldsymbol{\alpha}$ need to be allocated, where $\alpha_i \geq 0$ because this is a minimization constraint and we will assign a negative sign to the Lagrangian, so the multipliers need to be positive. If the Lagrange multiplier is zero, then this point plays no role in the classification. If it is greater than zero, then this data point is a support vector and plays a role in deciding the value of $\mathbf{w}$. Applying the KKT condition, the augmented cost function to be minimized is now given by

$$\begin{aligned} L_P &\equiv \frac{1}{2}||\mathbf{w}||^2 - \boldsymbol{\alpha}[y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1] \\ &\equiv \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{L} \alpha_i[y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1] \\ &\equiv \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{L} \alpha_i y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) + \sum_{i=1}^{L} \alpha_i \end{aligned}$$

The $\mathbf{w}$ and b that minimize L_P, need to be found. The above cost function is differentiated and set to zero.

$$\begin{aligned} \frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{L} \alpha_i y_i \mathbf{x}_i \\ \frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{i=1}^{L} \alpha_i y_i = 0 \end{aligned}$$

Those two equations are substituted into the previous one, and this new expression now needs to be maximized:

$$\begin{aligned} L_D &\equiv \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j \text{ subject to } \alpha_i \geq 0, \sum_{i=1}^{L} \alpha_i y_i = 0 \\ &\equiv \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i H_{ij} \alpha_j \text{ where } H_{ij} \equiv y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j \\ &\equiv \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \boldsymbol{\alpha}^\intercal \mathbf{H} \boldsymbol{\alpha} \text{ subject to } \alpha_i \geq 0, \sum_{i=1}^{L} \alpha_i y_i = 0 \end{aligned}$$

This new formulation L_D is called the dual form of the primary cost function L_P. It is important to note here that this formulation only needs the dot product of the input vectors and that the parameters to be determined in the dual form are the α_i rather than the $\boldsymbol{w}_i$. We need to maximize L_D. This is a convex quadratic optimization problem. If we run a QP solver, which will return $\boldsymbol{\alpha}$, then we can find $\boldsymbol{w}$ from the first constraint equation. Any support vector point satisfying the second constraint will have the form

$$y_s(\mathbf{w}^\intercal \mathbf{x}_s + b) = 1$$

and if we substitute into $\mathbf{w}$ the value of the first constraint we get

$$y_s(\sum_{m\in S}\alpha_m y_m \mathbf{x}_m^\intercal \cdot \mathbf{x}_s + b) = 1$$

S denotes the set of indices of the Support vectors. S is determined by finding the indices i where $\alpha_i > 0$, that is datapoints which lie exactly on margin and the corresponding constraint is active. Multiplying through by y_s and then using $y_s^2 = 1$

$$y_s^2(\sum_{m\in S}\alpha_m y_m \mathbf{x}_m^\intercal \mathbf{x}_s + b) = y_s \Rightarrow$$
$$b = y_s - \sum_{m\in S}\alpha_m y_m \mathbf{x}_m^\intercal \cdot \mathbf{x}_s$$

which provides the offset value. An average over all of the support vectors in S can also be taken

$$b = \frac{1}{N_s}\sum_{s\in S}(y_s - \sum_{m\in S}\alpha_m y_m \mathbf{x}_m^\intercal \cdot \mathbf{x}_s)$$

When data is not fully linearly separable, the first constraints are relaxed to allow for misclassified points. A positive slack variable ξ_i is introduced:

$$\mathbf{w}^\intercal \mathbf{x}_i + b \geq +1 - \xi_i \text{ for } y_i = +1$$
$$\mathbf{w}^\intercal \mathbf{x}_i + b \leq -1 + \xi_i \text{ for } y_i = -1$$

as before this is expressed better as:

$$y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1 + \xi_i \geq 0$$

In this soft margin SVM, points on the wrong side of the margin boundary receive an increasing with the distance penalty. The number and the intensity of misclassifications needs to be reduced, so our objective function needs to be adapted from previously so as to try and find the minimum of:

$$L = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{L}\xi_i \text{ subject to } y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1 + \xi_i \geq 0$$

The C hyperparameter controls the trade-off between the size of the margin and the slack variable penalty. This is reformulated as a Lagrangian as before and now the appropriate $\mathbf{w}$,b and ξ_i need to be found to minimize L_P.

$$L_P = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{L}\xi_i - \sum_{i=1}^{L}\alpha_i[y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) - 1 - \xi_i] + \sum_{i=1}^{L}\mu_i\xi_i$$

Differentiating w.r.t. $\boldsymbol{w}$, b, ξ_i and setting to zero we get:

$$\frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{L} \alpha_i y_i \mathbf{x}_i$$
$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{i=1}^{L} \alpha_i y_i = 0$$
$$\frac{\partial L_P}{\partial \xi_i} = 0 \Rightarrow C = \alpha_i + \mu_i$$

Substituting like before, this has the same form with one extra constraint, $\alpha \leq C$. b is then calculated just like before. (Fletcher, 2009)

4.3.2 SVM for Regression

Now, new unseen data points are not aimed to be classified into one of two categories but the aim is to predict a real-valued output for y. The form of our training data is the following:

$$\{\mathbf{x}_i, y_i\} \quad i = 1 \dots L, y_i \in \mathbb{R}, \ \mathbf{x}_i \in \mathbb{R}^D$$

It is understandable that regression SVM will need to use a more sophisticated penalty function. It needs to allocate a penalty only if the predicted values y_i is more than a distance ϵ away from the target value, denoted t_i. This can be expressed as:

$$|t_i - y_i| < \epsilon$$

The bounded region $y_i \pm \epsilon$ is an ϵ-insensitive tube. The other modification to the penalty function is that output variables which are outside the tube are given one of two slack variable penalties depending on whether they lie above (ξ^+) or below (ξ^-).

$$t_i \leq y_i + \epsilon + \xi^+$$
$$t_i \geq y_i - \epsilon - \xi^-$$

The error function for SVM regression can be written as

$$L = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{L}(\xi_i^+ + \xi_i^-)$$

This needs to be minimized subject to the constraints $\xi^+, \xi^- \geq 0$ plus the two constraints above. Lagrange multipliers $\alpha_i^+ \geq 0$, $\alpha_i^- \geq 0$, $\mu_i^+ \geq 0$, $\mu_i^- \geq 0$ are once again introduced.

$$L_P = C\sum(\xi_i^+ + \xi_i^-) + \frac{1}{2}||\mathbf{w}||^2 - \sum_{i=1}^{L}(\mu_i^+\xi_i^+ + \mu_i^-\xi_i^-) - \sum_{i=1}^{L}\alpha_i^+(\epsilon + \xi_i^+ + y_i - t_i) - \sum_{i=1}^{L}\alpha_i^-(\epsilon + \xi_i^- - y_i + t_i)$$

Substituting for y_i, differentiating w.r.t. $\mathbf{w}$, b, ξ^+, ξ^- and setting to zero:

$$\frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-)\mathbf{x}_i$$
$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-) = 0$$
$$\frac{\partial L_P}{\partial \xi_i^+} = 0 \Rightarrow C = \alpha_i^+ + \mu_i^+$$
$$\frac{\partial L_P}{\partial \xi_i^-} = 0 \Rightarrow C = \alpha_i^- + \mu_i^-$$

The first two equations are substituted into L_P and now L_D needs to be maximized w.r.t. α_i^+ and α_i^-.

$$L_D = \sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-)t_i - \epsilon\sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-) - \frac{1}{2}\sum_{i,j}(\alpha_i^+ - \alpha_i^-)(\alpha_i^+ - \alpha_i^-)\mathbf{x}_i^\intercal\mathbf{x}_j$$

plus the last two constraints.

Substituting the first constraint equation into $y_i = \mathbf{w}\cdot\mathbf{x}_i + b$, new predictions can be found using

$$y' = \sum_{i=1}^{L}(\alpha_i^+ - \alpha_i^-)\mathbf{x}_i^\intercal\cdot\mathbf{x}' + b$$

A set S of support vectors $\mathbf{x}_s$ can be created, by finding the indices i where $0 < \alpha < C$ and $\xi_i^+, \xi_i^- = 0$.

This gives rhe offset value:

$$b = t_s - \epsilon - \sum_{m\in S}^{L}(\alpha_m^+ - \alpha_m^-)\mathbf{x}_m^\intercal\mathbf{x}_s$$

and it's better to average over all indices i in S

$$b = \frac{1}{N_s}\sum_{s\in S}\left[t_s - \epsilon - \sum_{m\in S}^{L}(\alpha_m^+ - \alpha_m^-)\mathbf{x}_m^\intercal\mathbf{x}_s\right]$$

(Fletcher, 2009)

4.3.3 Nonlinear SVMs

When L_D was defined for the linearly separable data, a matrix $\mathbf{H}$ called the Grammian matrix was created from the dot product of our input variables.

$$H_{ij} = y_i y_j k(\mathbf{x}_i\mathbf{x}_j) = \mathbf{x}_i^\intercal\mathbf{x}_j$$

Unfortunately, data points are not always linearly separable.

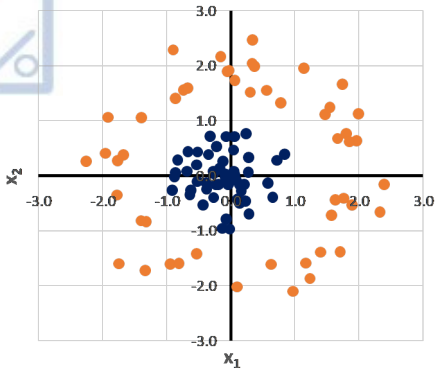


Figure 31: Non linearly separable data points

However, they may be able to be separated linearly if for example they are projected to a different and possibly higher dimensionality space. So for example if we perform a simple transformation to radial coordinates in the previous example, the data become linearly separable.

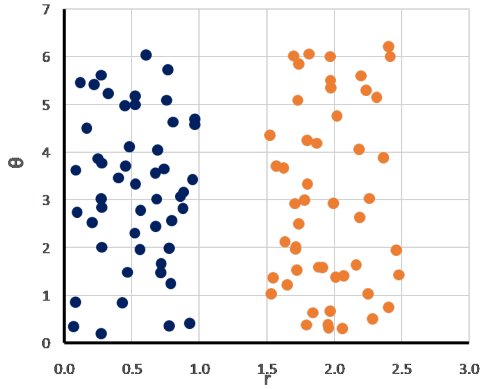


Figure 32: Data linearly separable in transformed space

The question now is how such a suitable transformation can be found in more complex case. Usually the answer is projecting in higher dimensions. Guessing the suitable transformation is not an easy task, nor is there a unique and correct way for it to be done. Furthermore, in higher dimensions data can't be visualized, so even if they are linearly separable it can't be seen to make certain. There comes the kernel trick. The main target is to project the data to a space $\mathbf{Z}$ of very high dimensionality. It can be easily shown that for many transformations, the dot product in this space $\mathbf{Z}$ can be computed in terms of the inner product in the original low dimensionality space $\mathbf{X}$. The computational complexity of an inner product is very low. So instead of transforming the feature space into a higher dimensional one, inner products of original data points can be calculated instead. Since the Gram matrix is only a function of the dot product, then an SVM regression in higher dimensions can be calculated easily without finding out what the exact transformation is, using only functions of the dot product called kernel functions.

$k(\boldsymbol{x}_i \boldsymbol{x}_j)$ is an example of a family of functions called Kernel functions. The function that was used in the linearly separable data is known as the linear kernel, or the simple dot product of the variables. Besides the linear kernel other popular ones include

- Radial Basis Kernel (RBF)

$$k(\mathbf{x}_i, \mathbf{x}_j) = e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}}$$

- Polynomial Kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^\intercal \cdot \mathbf{x}_j + \alpha)^b$$

- Sigmoidal Kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\alpha \mathbf{x}_i^\intercal \cdot \mathbf{x}_j - b)$$

(Fletcher, 2009) (Cortes & Vapnik, 1995) (Cristianini, Shawe-Taylor, et al., 2000) (Shawe-Taylor, Cristianini, et al., 2004)

For our thesis many SVMs were trained but ultimately a certain set of values, hyperparameters and kernels were decided to be better. These are the following

1. Linear kernel, $\epsilon = 0.001$, $C = 10$
2. Polynomial kernel, degree $= 2$, $\epsilon = 0.001$, $C = 10$
3. Polynomial kernel, degree $= 3$, $\epsilon = 0.001$, $C = 10$
4. Linear kernel, polynomial features of degree 2, $\epsilon = 0.001$, $C = 10$
5. RBF kernel, $\epsilon = 0.001$, $C = 1$
6. RBF kernel, polynomial features of degree 2, $\epsilon = 0.001$, $C = 1$
7. Grid optimized SVM
8. Grid randomly optimized SVM

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Linear	-0.95	4.98	3.22	3.92
Grid optimized	-0.17	4.58	2.75	3.67
Random Grid optimized	0.13	4.57	2.82	3.60
linear polynomial features	-0.14	6.01	3.16	5.11
RBF	-0.53	4.46	2.75	3.54
RBF polynomial features	-0.58	4.63	2.85	3.69
Polynomial 2nd degree	3.92	117.52	9.97	117.17
Polynomial 3rd degree	0.19	9.89	4.80	8.65

Figure 33: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Linear	-0.98	5.08	3.21	4.05
Grid optimized	0.01	4.64	2.98	3.56
Random Grid optimized	0.14	4.63	3.01	3.52
linear polynomial features	-0.55	5.16	3.06	4.20
RBF	-0.57	4.91	2.97	3.95
RBF polynomial features	-0.61	4.99	3.03	4.01
Polynomial 2nd degree	0.14	14.77	6.20	13.41
Polynomial 3rd degree	-0.46	7.85	4.81	6.22

Figure 34: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

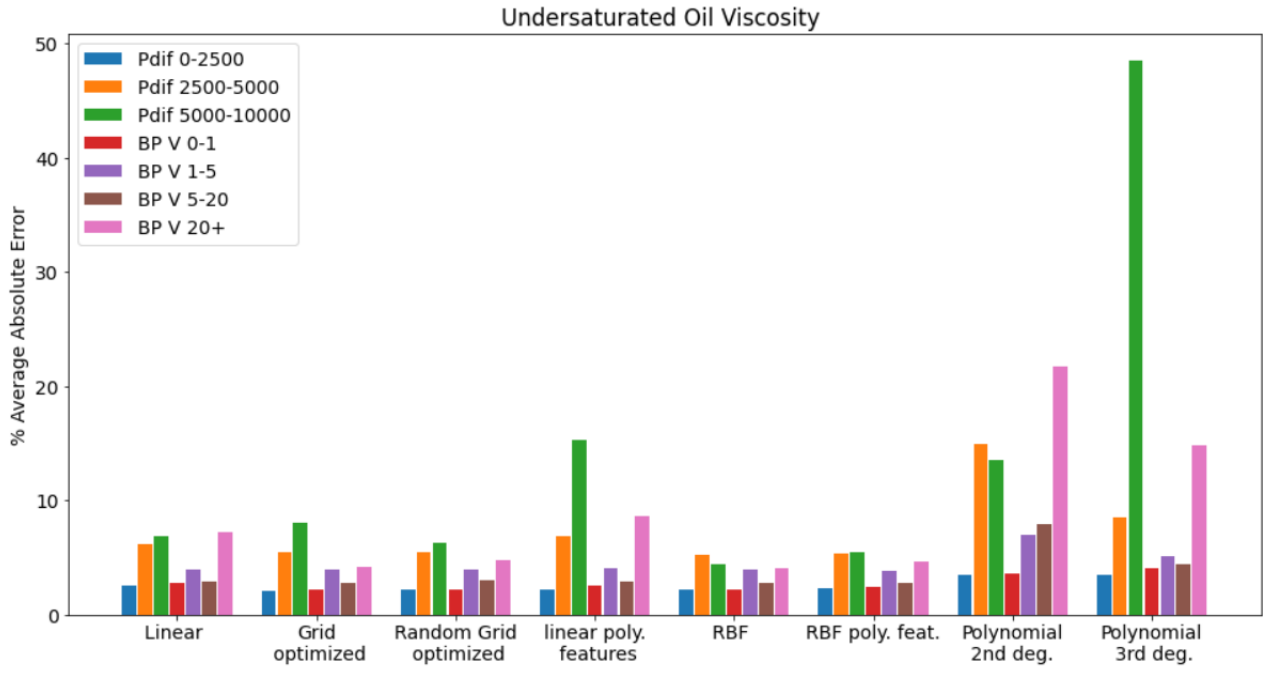


Figure 35: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

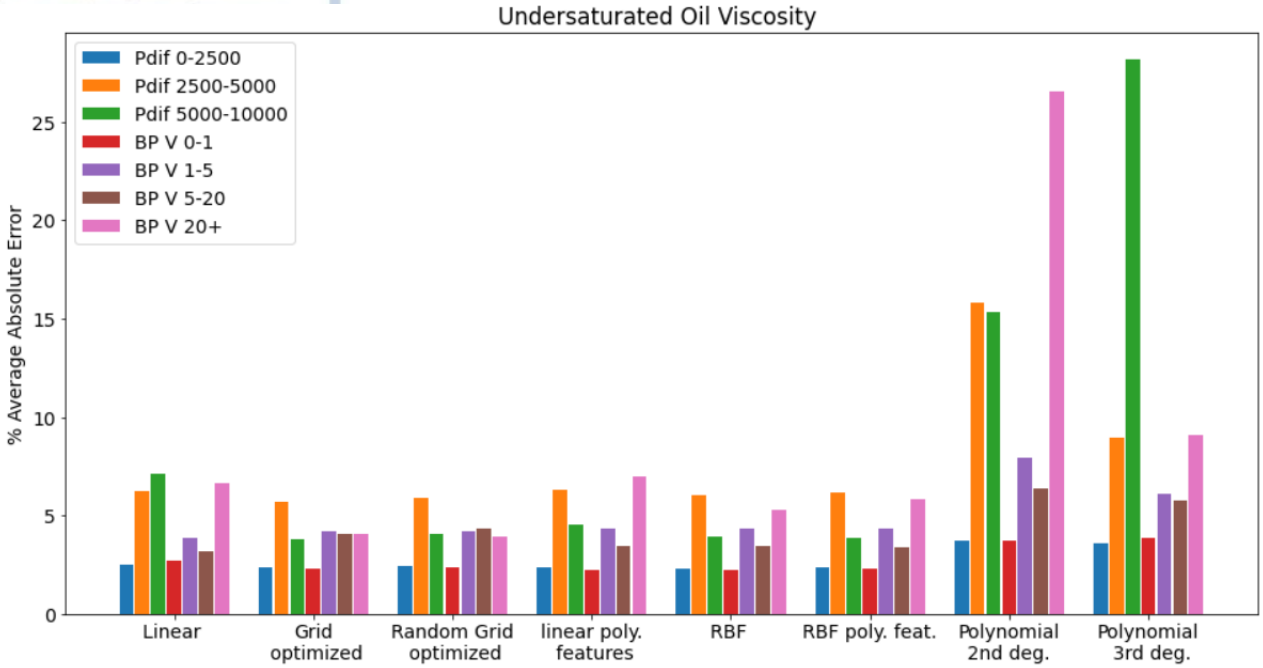


Figure 36: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

Once again it is clear that the best regression models are attained when using the full set of features. The best regression model trained was number five. The huge increase in error for big pressure differentials when using the polynomial kernel was very surprising. All in all results were as expected, with the lowest amount of error being distributed among fluids with bubble point viscosity values of less than 1 cp.

4.4 Decision Trees

Decision trees are powerful ML algorithms that can perform regression and classification tasks. Decision trees create nodes and predict a value in each node. So for example, if a prediction needs to be made on a new data point, the tree has to be traversed. The path starts at the root and then then follows a direction based on where the features' values lie. Finally a leaf node is reached that assigns a value to the instance based on the training instances that are associated with the leaf node. The decision trees trained, use the CART algorithm, that splits the training set in a way that minimizes MSE. The CART cost function is the following

$$J(k, t_k) = \frac{m_{left}}{m} MSE_{left} + \frac{m_{right}}{m} MSE_{right}$$

where m_{node} is the number of instances associated with the split, m is the total number of instances, k is the single feature that the splitting is done at any point, t_k is the threshold value, J is the cost function, $MSE_{\text{node}} = \sum_{i \in \text{node}} (\hat{y}_{\text{node}} - y^{(i)})^2$ and $\hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)}$.

Overfitting is usually a very common issue when decision trees are trained. This

can be limited by using a max depth regularization parameter. What this does is it limits the number of times that features can be split thus it shortens the nodes, hence the name maximum depth.

4.4.1 Random Forests

Random forest regressor, is an ensemble of decision trees. Ensemble methods are analyzed in the next subsection. Random forests are trained with the bagging method. Random forests train decision trees on random subsets of the training set. Sampling is performed with replacement.Then they simply aggregate the prediction of all trees. This algorithm searches for the best features when it splits a node, but it introduces a level of randomness to it. At the end there is greater tree diversity and a better model overall.

It needs to be said here that a method called grid search will be used. It was also used in the SVM section. A grid is basically a set of hyperparameter value pairs that are combined and the model is trained many times. A complete grid was used, meaning that every hyperparameter pair in our grid is selected and trained. A random grid was also used. A random grid is similar to a normal grid but, its hyperparameter values can lie in a spectrum rather than being discrete. Pairs are selected randomly so not every hyperparameter combination is tested making the search faster. The model obtained from the grid is then evaluated with k-fold cross validation.

K-fold cross validation is a procedure that is mainly used when the dataset is quite small so a separate validation set should not be created. What it does is it splits randomly the training set into k separate groups called folds. Then for each group it takes it as a holdout validation set and it trains the model on the other (k-1). It retains every score and aggregates the result. (Géron, 2019)

For the purposes of this work the following models were trained

1. Decision Tree without any regularization
2. Random forest with 100 tree estimators
3. Random forest from Grid search
4. Random forest from random Grid search
5. Decision tree with polynomial features of degree=2
6. Random forest of 100 decision trees with polynomial features of degree=2

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Decision Tree	0.79	7.53	4.53	6.06
Random Forest	0.03	4.62	2.79	3.68
Grid optimized forest	0.12	4.53	2.82	3.55
Random Grid optimized forest	0.15	4.51	2.79	3.54
Decision Tree polynomial features	0.75	7.36	4.64	5.77
Random Forest polynomial features	0.03	4.54	2.77	3.60

Figure 37: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Decision Tree	0.52	7.55	4.66	5.97
Random Forest	-0.04	4.66	2.93	3.62
Grid optimized forest	0.09	4.66	2.94	3.62
Random Grid optimized forest	0.07	4.69	2.94	3.65
Decision Tree polynomial features	0.80	7.84	4.83	6.23
Random Forest polynomial features	0.09	4.74	2.99	3.68

Figure 38: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

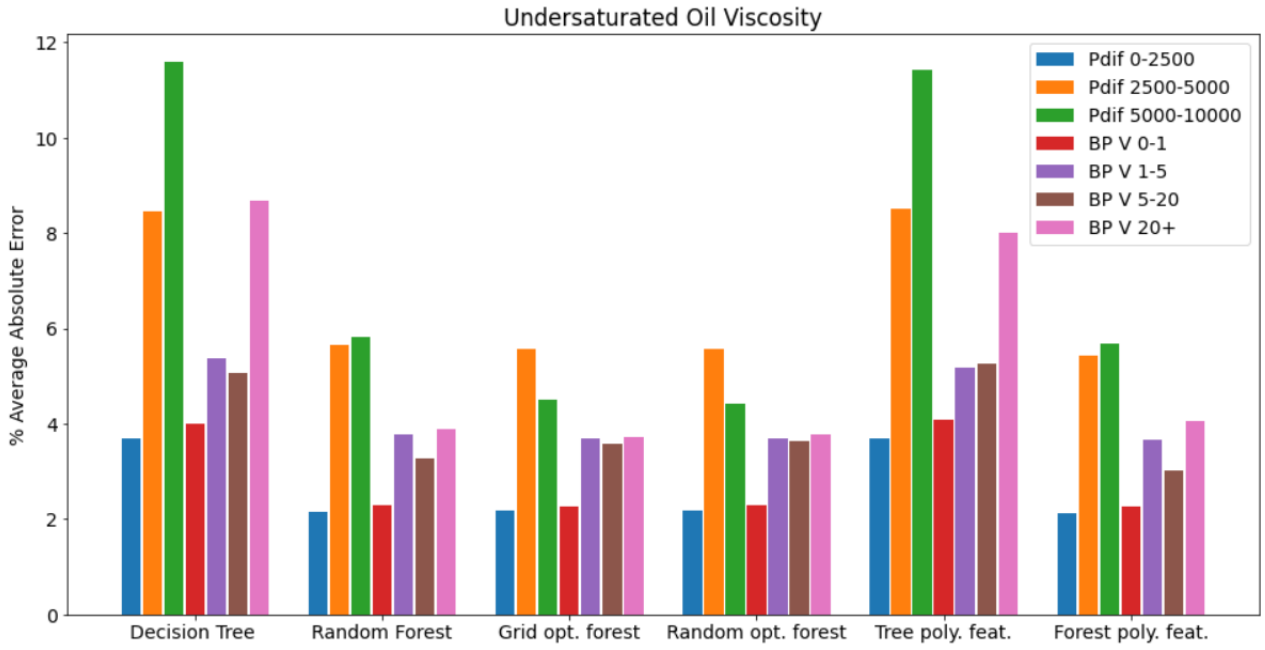


Figure 39: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

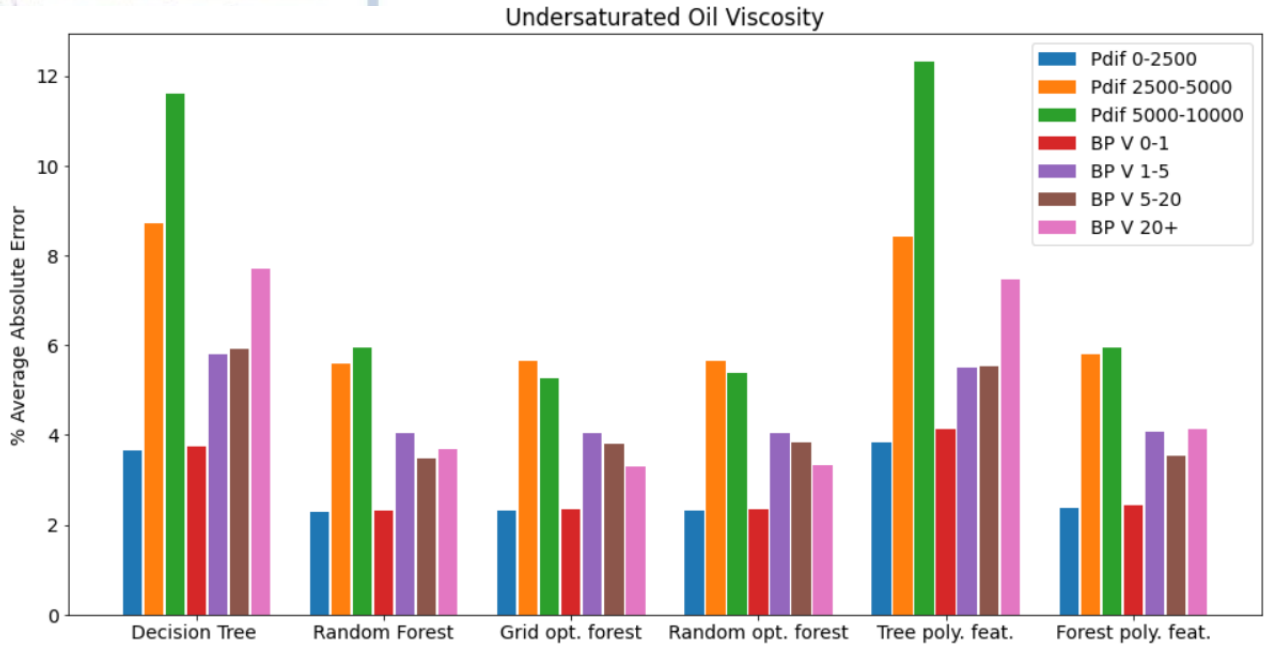


Figure 40: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

Again the best results occur when the full feature set is used. The best regression model is number two, the simple Random Forest. The worst is the simple decision tree. The important issue to be concluded here is how an average regression model (Decision Tree) can become so strong (Random Forest) using ensemble learning.

4.5 Ensemble Regression models

Ensemble regression models involve the combination of regression models. Random forest is a form of an ensemble regressor.

4.5.1 Voting

Voting aggregates the predictions of many regression models of our choosing and predict the value that is the average of the contributing models. Voting works great even with weak learners, supposing that they are independent and that they produce uncorrelated errors. We created two voting regression models, using the RBF SVM from before and the random forest. One regression model used the normal features, and the other used the polynomial ones.

4.5.2 Bagging

This method is a generalized random forest method. Contrary to the random forest where the base estimator is a decision tree, Bagging can take any method of our choosing as a base estimator. So how it works is already mentioned. One was trained here with 200 estimators (Decision trees), with max sampling size of 5,000 and replacement.

4.5.3 AdaBoostRegressor

Boosting is a method that combines several weak learners into a strong one. The idea is to train models sequentially, and each models tries to correct its predecessor. AdaBoost is one of the most popular boosting methods. Every new predictor pays more and more attention to the instances that the predecessor underfitted. New predictors focus more and more on the hard cases. Once all predictors are trained, the ensemble makes predictions like bagging.

Initially every instance weight w_i^1 is set to $\frac{1}{N}$ where N is the total number of instances in the sample and $i = 1, 2, \dots N$. For $j = 1, 2, \dots, T$ or while $\bar{L}^j$ defined below is less or equal to 0.5. A sample of size N is drawn from the data with replacement and with probability w_n^j. A weak learner j is fitted to the resampled data and the fitted values are calculated on the original dataset. These values are denoted with $f^j(\mathbf{x}_i)$. The observation error L_i^j is calculated.

$$L_i^j = \frac{|y_i - f^j(\mathbf{x}_i)|}{\max\{|y_i - f^j(\mathbf{x}_i)|\}}$$

The model error is then calculated.

$$\bar{L}^j = \sum_{i=1}^{N} L_i^j w_i^j$$

If the model error is greater than 0.5 the iteration is stopped at the $j-1$ predictor. Define:

$$\beta^j = \frac{\bar{L}^j}{1 - \bar{L}^j}$$

The lower the β^j the greater the confidence in our model. The model weights are updated as:

$$w_i^{j+1} = \frac{w_i^j (\beta^j)^{1-L_i}}{\sum_{i=1}^{N} w_i^j (\beta^j)^{1-L_i}}$$

which increases the weights for observations with a grater error. Finally the overall fitted value for observation i is set to the weighted median of $f^j(\mathbf{x}_i)$ using weights $\log(1/\beta^j)$ for model j.

For our purposes two hundred decision trees were trained for our AdaBoost regressor.

4.5.4 Gradient Boosting

Gradient boosting works by adding predictors sequentially to an ensemble. Each new predictor corrects its predecessor. However, unlike Adaboost that tweaks the instance weights at every iteration, this method fits the new predictor to the residual errors of the previous one.

A gradient boost regression model based on tree regression with a max depth of

16 and a learning rate of 0.1 was trained in our work. LightGBM, a popular gradient boosting library that also uses tree regression was also utilized to create another gradient boosting regression model. (Géron, 2019)

4.5.5 Stacking

Stacking is an ensemble method, that instead of using functions like voting to aggregate the predictions of all predictors in an ensemble, it trains a model to perform this task. In our case the same estimators as in the voting model were used, but the aggregation estimator was set to a random forest regressor.

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Voting polynomial features	-0.29	4.44	2.71	3.52
Voting	-0.29	4.39	2.68	3.49
Bagging	-0.04	4.47	2.73	3.54
AdaBoost	0.12	4.94	2.92	3.99
Gradient Boosting	0.20	4.80	2.96	3.78
Light	-0.18	4.72	3.03	3.62
Stacking	0.20	5.17	3.24	4.02

Figure 41: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Voting polynomial features	-0.30	4.70	2.89	3.73
Voting	-0.34	4.63	2.85	3.67
Bagging	0.00	4.65	2.91	3.62
AdaBoost	0.00	5.15	3.21	4.03
Gradient Boosting	0.28	5.28	3.30	4.13
Light	-0.21	4.74	3.01	3.67
Stacking	0.06	5.06	3.22	3.90

Figure 42: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

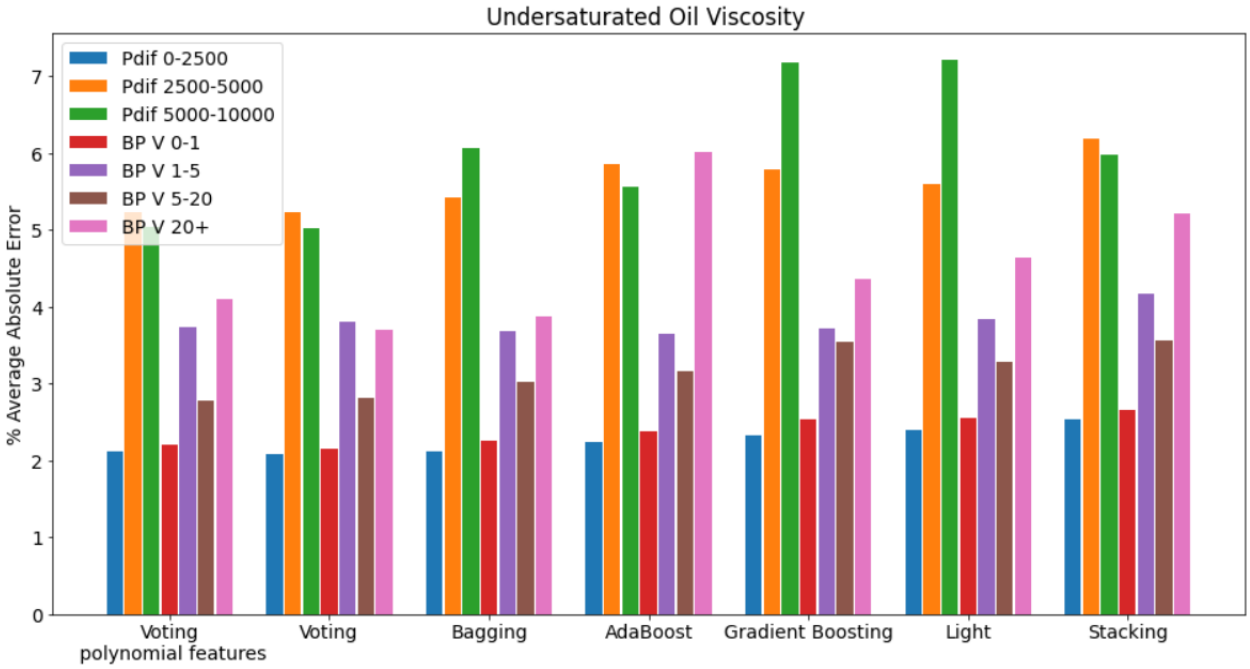


Figure 43: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

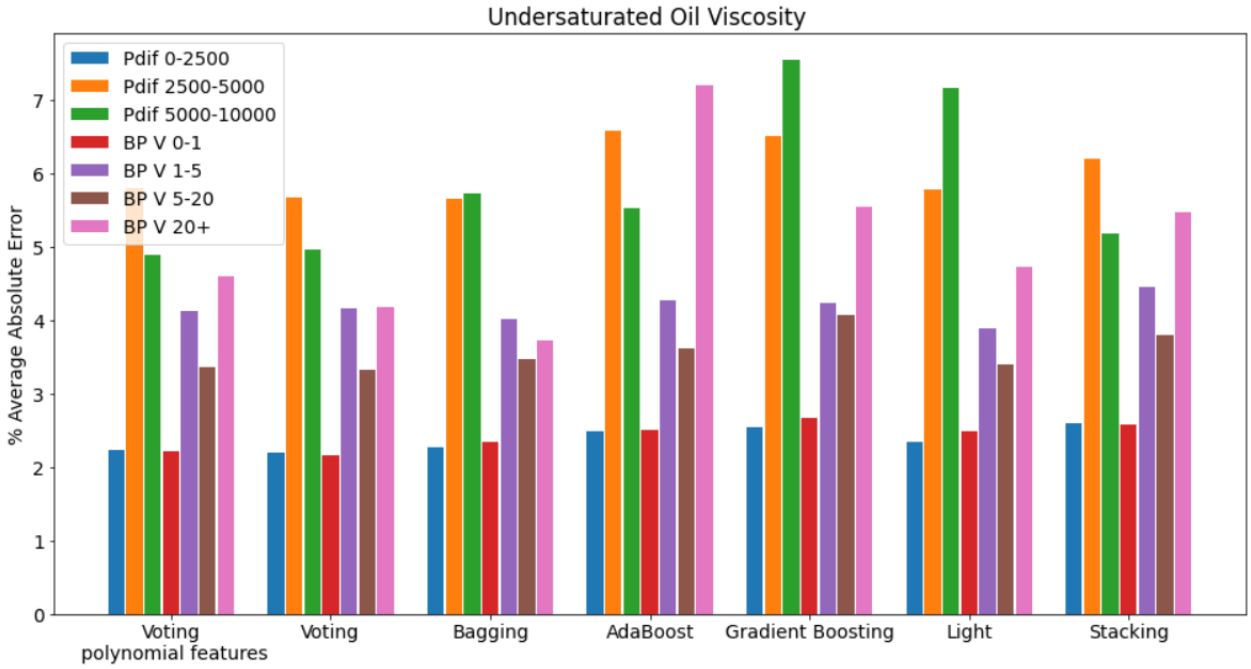


Figure 44: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

Once again it's clear that the full feature set produces the best results. Almost all ensemble regression models produce great results with the best of the bunch being the voting regression model.

4.6 Neural Networks

Neural networks are machine learning algorithms with vast applications and diverse architectures. Things will be kept simple in this work and only artificial neural networks (ANNs) will be considered whereas deep learning algorithms will not be examined. Our work involves a simple regression task after all, there is no need to go into deep learning. The Perceptron architecture is introduced, which is the one that Tensorflow uses. It is based on an artificial neuron called a threshold logic unit (TLU). The inputs and output are scalars, and each input connection is associated with a weight. The TLU computes a weighted sum of its inputs, then applies an activation function to that sum and outputs a result. A Perceptron is simply composed of a single layer of TLUs with each TLU connected to all inputs. When all neurons in a layer are connected to every neuron in the previous layer, then this layer is called dense or fully connected. All the input neurons form the input layer. Moreover, an extra bias feature is generally added. It is typically represented using a special type of neuron called a bias neuron, which outputs 1 all the time.

It is possible to efficiently compute the output of a layer of artificial neurons for several instances at once.

$$h_{\mathbf{W},\mathbf{b}}(\mathbf{X}) = \phi(\mathbf{XW} + \mathbf{b})$$

With $\mathbf{X}$ being the matrix of input features, $\mathbf{W}$ the matrix of weights and $\mathbf{b}$ is the bias vector. ϕ is the activation function. The perceptron is trained using a variation oh Hebb's rule "cells that fire together, wire together". This takes into account the error made by the network when it makes a prediction and reinforces connections that try to reduce the error. Specifically we feed it with one instance at a time, and for each instance it makes its predictions. For every output neuron that produced a wrong prediction, it reinforces the connection weights from the inputs that would have contributed to the correct prediction.

$$w'_{i,j} = w_{i,j} + \eta(y_j - \hat{y}_j)x_i$$

η is the learning rate, x_i is the i^{th} input value of the current training instance, $w_{i,j}$ is the connection weight between the i^{th} input neuron and the j^{th} output neuron, $\hat{y}_j$ is the output neuron for the current training instance and y_j is the target output of this output neuron for the current training instance. The decision boundary of each output neuron is linear, so Perceptrons are incapable of learning complex patterns. However, if the training instances are linearly separable, this algorithm would converge to a solution. This is called the Perceptron convergence theorem.

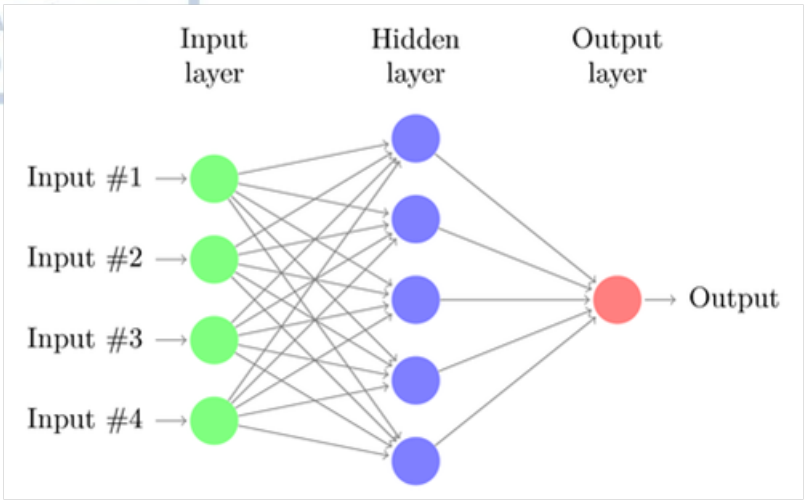


Figure 45: Typical MLP

Some limitations of Perceptors can be eliminated by stacking multiple Perceptors. The resulting ANN is called Multilayer Perceptron (MLP). Its composition is an input layer, one or more layers of TLUs called hidden layers, and one final layer of TLUs (in or case for regression it's just one TLU) called the output layer. (Géron, 2019)

In order to train an MLP the backpropagation training algorithm is used. Essentially, this is gradient descent with an efficient technique for computing gradients automatically. In just a forward and a backward pass through the network, the algorithm is able to compute the gradient of the network's error with regard to every single model parameter. The result is that it is able to find how each bias and weight need to be tweaked in order to minimize the error. Once it has these gradients, a regular gradient descent step is performed, and the whole process is repeated until convergence. The complete algorithm will now be shown.

1. The data is split in batches (mini-batches) and the algorithm goes through the full training set (epoch) multiple times.

2. Each mini-batch is passed to the input layer. Subsequently this sends it to the first hidden layer. The algorithm then computes all the outputs in this layer for every instance in the mini-batch. The result is passed on to the next layer and so on until we get the output of the last layer. This is the forward pass.

3. The algorithm measures the network's output error using a loss function of our choice.

4. It then computes how much each output connection contributed to the error. This is done by applying the chain rule.

5. The algorithm then measures how much of these error contributions came from each connection in the layer below working backward until the algorithm reaches the input layer. The reverse pass efficiently measures the error gradient

across all the connection weights in the network by propagating the error gradient backward through the network

6. It finally performs gradient descent using the error gradients it just computed.

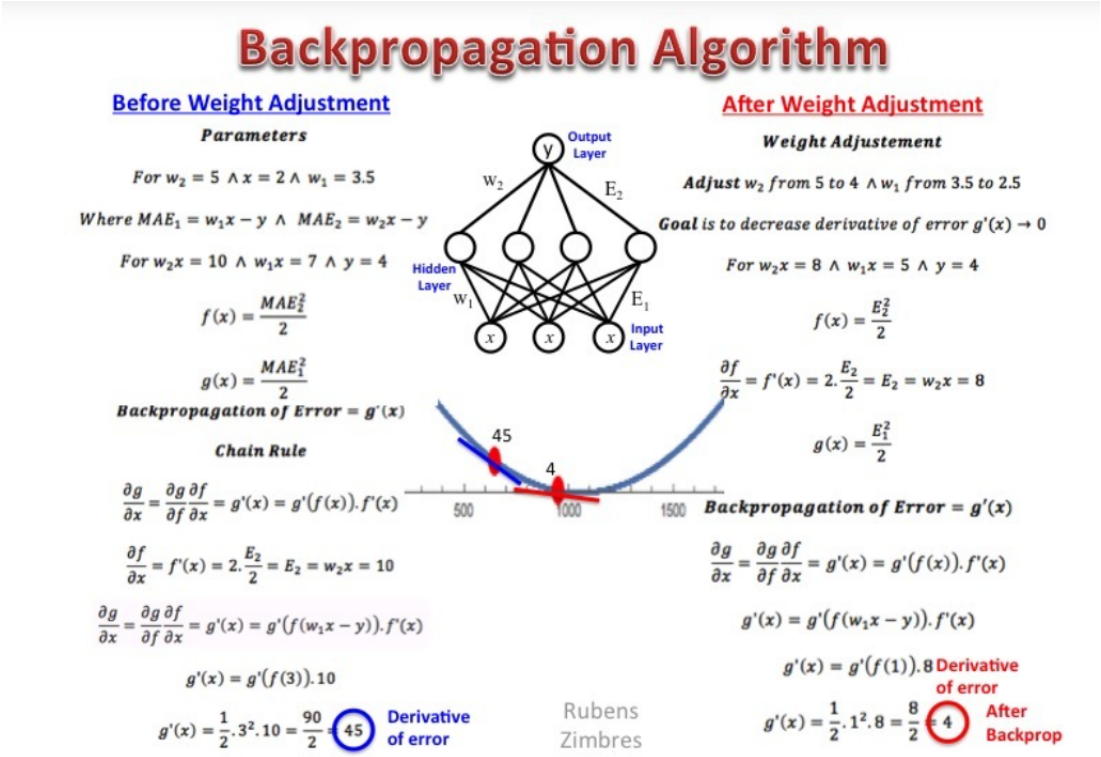


Figure 46: The algorithm summarized in one picture

(Rojas, 1996)
For the algorithm to work properly it was important to replace the step activation function, into a continuous one like the sigmoid function, or the Rectified Linear Unit function etc that have a gradient everywhere.

MLPs can be used for regression. If a single value is to be predicted, then just one output neuron is needed. In general no activation function for the activation layer for the output neuron should be used. In our case where the prediction of only positive values is demanded, the ReLU activation function can be used in the output layer.

For our purposes Neural networks with one, two, three and four hidden layers were trained. A first layer of five hundred neurons was always initialized and then the number decreased progressively at every next layer. The ReLU activation function , a batch size of 16, twenty epochs and the Adam optimizer algorithm were used. the ReLU is defined as follows:

$$R(z) = \max(0, z)$$

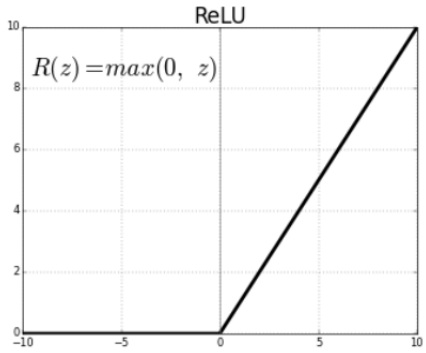


Figure 47: The rectified linear activation function

Adam (Kingma & Ba, 2014) is an optimization algorithm used instead of stochastic gradient descent. It is a combination of the gradient descent with momentum algorithm and the RMSP algorithm. The momentum algorithm is used to accelerate the gradient descent algorithm by taking into consideration the exponentially weighted average of the gradients. Using averages makes the algorithm converge towards the minimum in a faster pace.

$$w_{t+1} = w_t - \alpha m_t$$

where

$$m_t = \beta m_{t-1} + (1 - \beta) \left[\frac{\partial L}{\partial w_t} \right]$$

m_t is the aggregate of the gradients at time t, w_t are the weights at time t, α is the learning rate, L is the Loss function and β is the moving average parameter.

Root mean square prop or RMSP is an adaptive learning algorithm that takes the exponential moving average.

$$w_{t+1} = w_t - \frac{\alpha_t}{(u_t + \epsilon)^{0.5}} \cdot \left[\frac{\partial L}{\partial w_t} \right]$$

where

$$u_t = \beta u_{t-1} + (1 - \beta) \cdot \left[\frac{\partial L}{\partial w_t} \right]^2$$

u is the sum of square of past gradients and ϵ is a small positive constant.

Adam Optimizer inherits the positives of the above two methods and builds upon them to give a more optimized gradient descent. It is able to oscillate very little close to the global minimum while taking big steps to pass local minimum.

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
1 hidden layer	-0.05	4.63	2.71	3.76
2 hidden layers	-1.21	5.75	3.35	4.83
3 hidden layers	-0.08	7.01	3.61	6.01
4 hidden layers	1.32	7.50	3.68	6.67

Figure 48: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
1 hidden layer	-1.05	4.58	2.79	3.78
2 hidden layers	-0.66	4.70	2.82	3.82
3 hidden layers	-0.57	4.83	2.85	3.94
4 hidden layers	-1.11	4.89	2.97	4.03

Figure 49: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

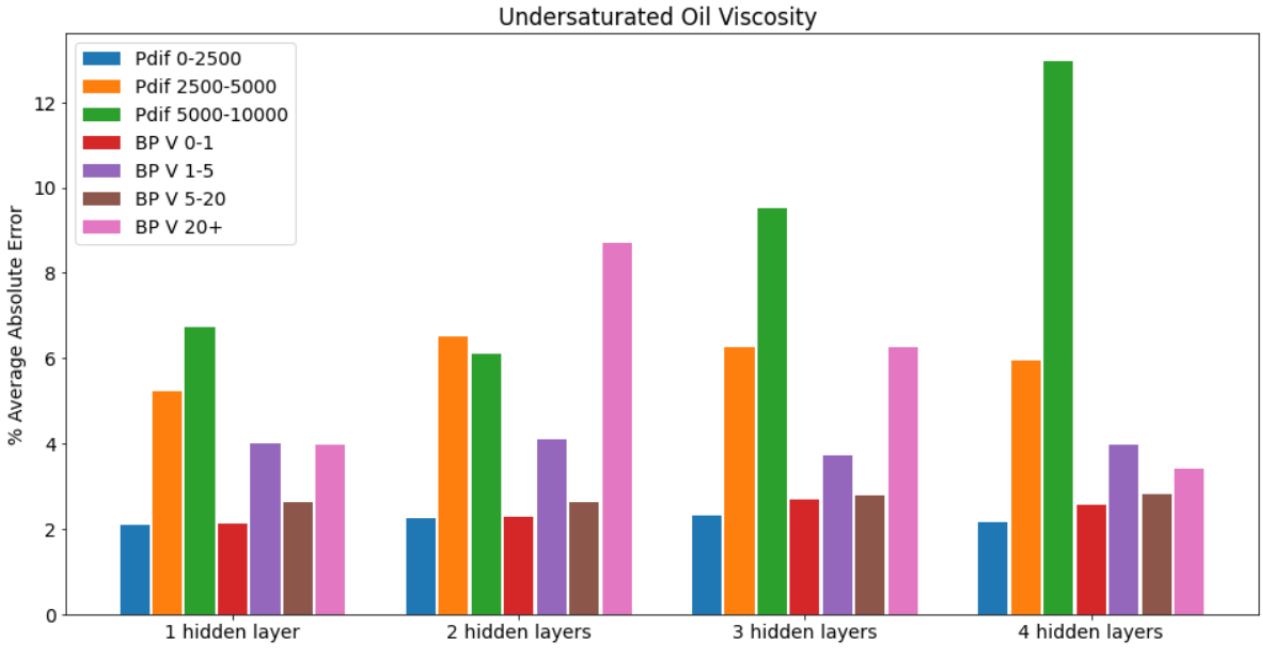


Figure 50: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

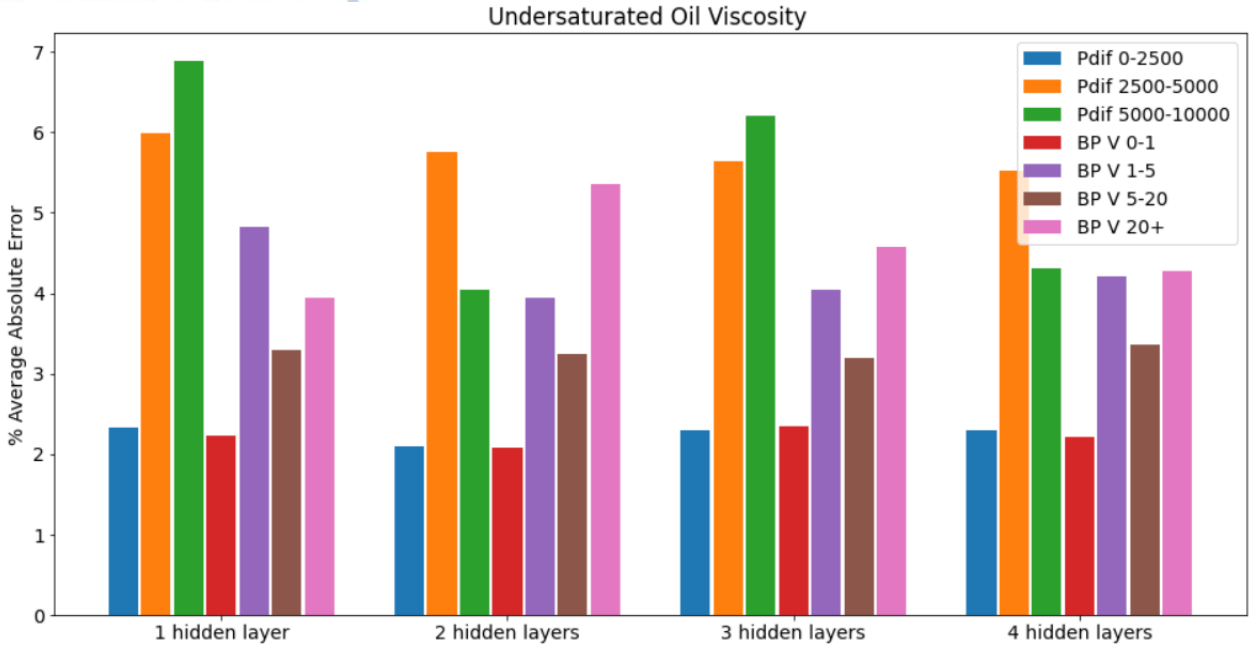


Figure 51: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

The best regression model is the one with one hidden layer. Perhaps more complex nets are not needed in order to produce a better result. It seems as though no deep information is to be revealed in this problem. It is also surprising that the partial features work better on nets with more hidden layers. The key point here is that a net of just one hidden layer produces a regression model that is of similar performance to the best ones from before.

4.7 Symbolic Regression

Symbolic regression is a genetic programming machine learning technique that tries to identify the mathematical expression that best describes a relationship. It starts by building a population of naive random formulas to find relationships between independent variables and their dependent variable targets in order to predict new data. It uses mathematical relations of our choosing to do so. Each successive generation of formulas is evolved from the previous one by selecting the fittest (we need to define a fitness function) individuals from the population. Then the fittest individuals undergo genetic operations. Genetic programming is a stochastic optimization process. Random individuals among the fittest ones from the current generation are selected to undergo changes in order to enter the next generation. This happens every time an initial population is conceived and with every step in the process. Symbolic regression can be represented in many ways, like a LISP symbolic expression that uses prefix notation (e.g. $y = (+(-(\times X_0 X_0)(\times 3 X_1))0.5))$ or even in tree form. The term subtree will be used below to refer to any tree node or part of a node that is going to be a part of mutation.

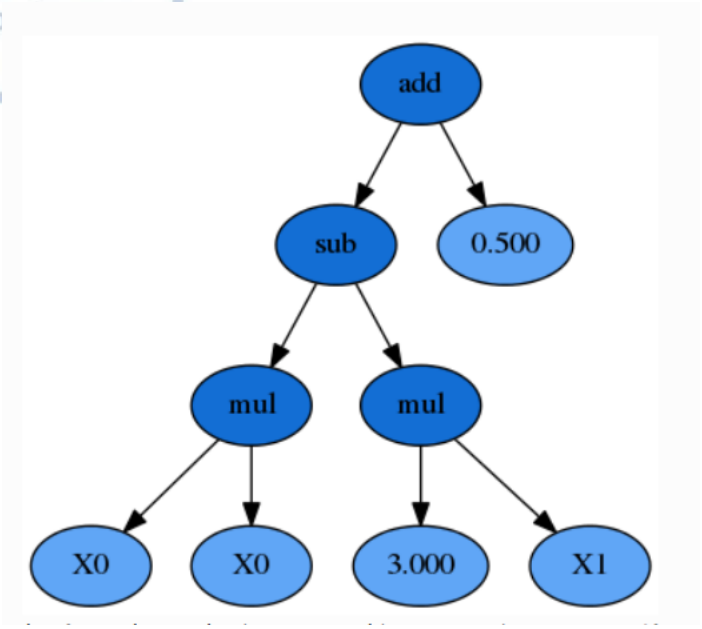


Figure 52: tree form of symbolic regression

The set of arithmetic operators that the algorithm will try were chosen. In our case addition, subtraction, multiplication, division were used.

In order to determine how well an algorithm performs a fitness function is used. Mean absolute percent error was used to evaluate our symbolic regression algorithms. It is also important for the features to be in the same order of magnitude with the target.

In our case a population size of 5,000 was chosen. This is the number of function competing in every generation. In order to decide which ones get to evolve in the next generation, tournaments are hosted. Smaller subsets from the population are selected at random to compete. Large tournaments will find fitter expressions quicker and the evolution process will tend to converge to a solution in less time. Smaller tournaments will maintain more diversity in the population as more formulas are given a chance to evolve. Evolution is using the fitness function to find the fittest individual from the tournament to survive. This individual does not just graduate to the next generation unchanged. There are several evolution parameters to be chosen, each with a specified probability.

Now the evolution parameters have to be chosen. One of them is crossover (probability=0.8), the principle mixing method of genetic material between individuals of the generation. A winner of a tournament is selected and a random subtree is replaced by the winner of a second tournament who is called the donor. The donor also has a subtree selected at random, then this is inserted into the original parent to form an offspring in the next generation.

Another evolution parameter is subtree mutation (probability=0.1). This is an

aggressive parameter because more genetic material will be replaced by naive and random components. It is good to maintain diversity. Just like before the winner of a tournament has a subtree replaced but at random this time.

Hoist mutation (probability=0.05) is another parameter that removes material from tournament winners. Point mutation (probability=0.05) takes the winner of a tournament and replaces random nodes from it. Terminals are replaced by other terminals and functions are replaced by other functions that require the same number of arguments as the original node. The resulting tree forms an offspring in the next generation. (*Introduction to GP¶*, n.d.)

Our feature selection had to be changed for this regression model to be trained well. Taking notes from the literature and it was clear that Beal's model is the one that fits our data the best. So its success was aimed to be repeated by finding regression models of the form:

$$\mu \propto \mu_{ob} \cdot (\Delta P)^{\alpha} \cdot f(P_b, P, GOR, API, T)$$

After many runs 5 respectable models were found, that perform worse than our previous regression models, but have the advantage of having an explicit mathematical formula unlike other machine learning models.

Model 1

$$\mu = \mu_{ob}(1.133 + 0.114 \cdot \mu_{ob}^{*})$$

Model 2

$$\mu = \mu_{ob}(1.091 + 0.011(2\mu_{ob}^{*} + 0.977)(\mu_{ob}^{*} + 2.931) + 0.011P_b^{*}(3.045 + 0.011(2\mu_{ob}^{*} + 0.977)(\mu_{ob}^{*} + 2.931) - 0.915 * GOR^{*}))$$

Model 3

$$\mu = \mu_{ob}\left(1.168 - \frac{0.126 * GOR^{*}}{\frac{0.126 - Tr^{*}}{(-0.842 - P^{*} \cdot \Delta P^{*} + P^{*})(-0.842 - Tr^{*} \cdot \Delta P^{*})}} - 1.089\right)$$

Model 4

$$\mu = \mu_{ob}(1.132 + 0.098 * GOR^{*})$$

Model 5

$$\mu = \mu_{ob}(1.143 + 0.066 * \mu_{ob}^{*} + 0.034 * GOR^{*} + (0.034 * P_b^{*} + 1.143) * 0.034 * \mu_{ob}^{*})$$

where $P_b^{*} = (\Delta P^{0.85}/\ln(P_b{+}60){-}69)/2494.64$, $P^{*} = (\Delta P^{0.85} * P^{-0.2} {-} 95.28)/3728.77$, $\Delta P^{*} = (\Delta P^{0.85} - 502.25)/122469.22$, $\mu_{ob}^{*} = (\Delta P^{0.85} * \mu_{ob}^{0.15} - 523.60)/159608.29$, $GOR^{*} = (\Delta P^{0.85}/\ln(GOR{+}20){-}84.27)/3934.46$, $Tr^{*} = (\Delta P^{0.85} * Tr^{-1} {-} 1.36)/0.87$.

This unorthodox form of the features is due to the scaling done in order for the genetic algorithm to work without issues.

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
1	-1.28	6.02	3.82	4.83
2	-1.19	5.97	3.68	4.85
3	-0.21	6.49	4.16	4.99
4	-1.25	6.35	3.93	5.14
5	-0.33	6.08	3.81	4.75

Figure 53: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

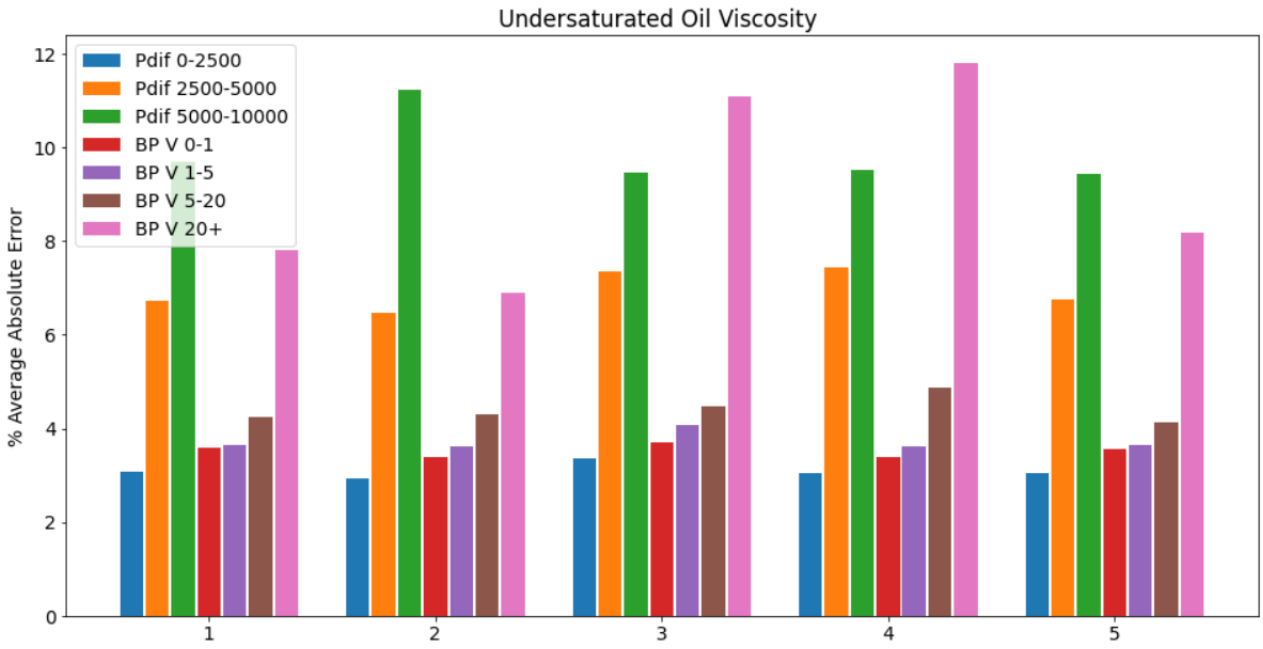


Figure 54: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

These results are not being shown on the test set, but the whole dataset ($\mu_{ob} < 50$) was used to evaluate. Since the formulas are mostly linear, there was no fear of overfitting, so the whole set is used for evaluation. The results are not bad, but they pale in comparison to Beal's correlating equation.However they perform better on the dataset than most correlations that utilize pressure difference and viscosity ratio. No symbolic regression model will be selected for the comparison phase.

5 Comparisons

In this chapter the best regressor from every category will be chosen (except symbolic regression) along with the two best literature correlations in order to compare them and find out if our models present an improvement over the literature. It is already mentioned that just because AARE was lower in our trained regressors, it doesn't automatically mean that they are better fitted for the prediction of undersaturated oil viscosity. Literature was judged on the whole dataset where μ_{ob} was below 50 cp, while our models where judged on a small subset of it called the test set. What will now be done instead is that the original dataset will be split again using five different random states, retrain our best regressors every time, predict the values on the new test set and finally predict the values of the best literature models on the test set as well.

The regressors chosen are the following:

- Ridge Regression
- SVM with RBF kernel
- Random Forest
- the voting regressor that votes among the SVM and the Random Forest
- ANN with 1 hidden layer and 500 neurons

For the literature correlations "Beal" and" Kouzel modified" were chosen as the representative when $\mu = f(\mu_{ob}, \Delta P, P, P_b)$ and "Almehaideb" and "Dindoruk and Christman" when $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$. The results of the five runs will now be shown.

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-1.20	5.69	3.61	4.55
SVM RBF kernel	-1.67	5.71	3.53	4.79
Random Forest	-1.17	5.74	3.49	4.71
Voting Regressor	-1.46	5.64	3.45	4.69
ANN with 1 layer	-1.69	5.36	3.34	4.52
Almehaideb	2.24	10.40	6.63	8.32
Dindoruk and Christman	4.06	8.71	6.29	7.27

$\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.96	5.97	3.71	4.77
SVM RBF kernel	-1.44	5.82	3.60	4.79
Random Forest	-0.79	5.82	3.68	4.57
Voting Regressor	-1.14	5.71	3.55	4.61
ANN with 1 layer	-1.64	5.54	3.47	4.62
Beal	-2.38	5.60	3.70	4.84
Kouzel	-2.52	6.65	4.30	5.67

$\mu = f(\mu_{ob}, \Delta P, P, P_b)$

first run

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	0.29	5.11	3.25	3.96
SVM RBF kernel	-0.23	4.67	2.94	3.64
Random Forest	0.39	4.84	3.07	3.76
Voting Regressor	0.07	4.66	2.90	3.65
ANN with 1 layer	-0.27	4.57	2.90	3.54
Almehaideb	3.02	9.01	6.20	7.20
Dindoruk and Christman	6.06	13.32	7.00	12.85

$\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	0.22	5.07	3.30	3.85
SVM RBF kernel	-0.30	4.94	3.15	3.81
Random Forest	0.20	5.08	3.30	3.86
Voting Regressor	-0.08	4.92	3.14	3.79
ANN with 1 layer	-0.71	4.89	3.07	3.87
Beal	-1.40	5.04	3.20	4.14
Kouzel	-0.83	5.49	3.66	4.18

$\mu = f(\mu_{ob}, \Delta P, P, P_b)$

second run

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.48	5.39	3.32	4.27
SVM RBF kernel	-1.07	5.02	3.15	4.06
Random Forest	-0.30	5.06	3.10	4.01
Voting Regressor	-0.71	4.94	3.03	3.97
ANN with 1 layer	-0.73	5.28	3.25	4.23
Almehaideb	1.18	8.82	5.84	6.71
Dindoruk and Christman	6.33	17.80	8.39	16.93

$\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.56	5.69	3.44	4.56
SVM RBF kernel	-1.23	5.49	3.34	4.52
Random Forest	-0.24	5.45	3.41	4.26
Voting Regressor	-0.78	5.33	3.24	4.30
ANN with 1 layer	-0.95	5.51	3.34	4.48
Beal	-1.91	5.48	3.52	4.62
Kouzel	-1.96	6.36	3.93	5.36

$\mu = f(\mu_{ob}, \Delta P, P, P_b)$

third run

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.47	5.47	3.63	4.12
SVM RBF kernel	-0.76	5.24	3.48	3.98
Random Forest	-0.44	5.08	3.34	3.85
Voting Regressor	-0.61	5.06	3.37	3.83
ANN with 1 layer	-2.00	5.24	3.55	4.34
Almehaideb	2.45	9.94	6.40	8.00
Dindoruk and Christman	5.40	12.15	6.92	11.36

$\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.52	5.04	3.39	3.77
SVM RBF kernel	-1.02	4.99	3.35	3.84
Random Forest	-0.41	5.28	3.47	4.00
Voting Regressor	-0.74	5.01	3.29	3.85
ANN with 1 layer	-1.85	4.87	3.41	3.94
Beal	-1.70	5.04	3.35	4.13
Kouzel	-1.80	5.66	3.96	4.43

$\mu = f(\mu_{ob}, \Delta P, P, P_b)$

fourth run

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.58	7.05	4.17	5.72
SVM RBF kernel	-0.96	6.82	4.00	5.61
Random Forest	-0.84	6.78	4.05	5.50
Voting Regressor	-0.92	6.67	3.95	5.46
ANN with 1 layer	-1.57	6.93	4.01	5.87
Almehaideb	2.42	16.23	7.58	14.56
Dindoruk and Christman	4.17	8.63	6.49	7.05

$\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Ridge Regression	-0.96	6.84	4.07	5.58
SVM RBF kernel	-1.32	6.69	4.00	5.52
Random Forest	-0.85	6.79	4.19	5.41
Voting Regressor	-1.11	6.64	4.02	5.40
ANN with 1 layer	-0.81	6.76	4.16	5.40
Beal	-2.54	6.71	4.26	5.77
Kouzel	-2.08	7.39	4.55	6.18

$\mu = f(\mu_{ob}, \Delta P, P, P_b)$

fifth run

It seems that our regressors are better but that is not always the case. Beal's correlation is the only one that can compete with our models in terms of performance. More shuffles of our data need to be provided to make a complete case. Instead of choosing five random states to shuffle, one hundred will now be used. The following graphs result.

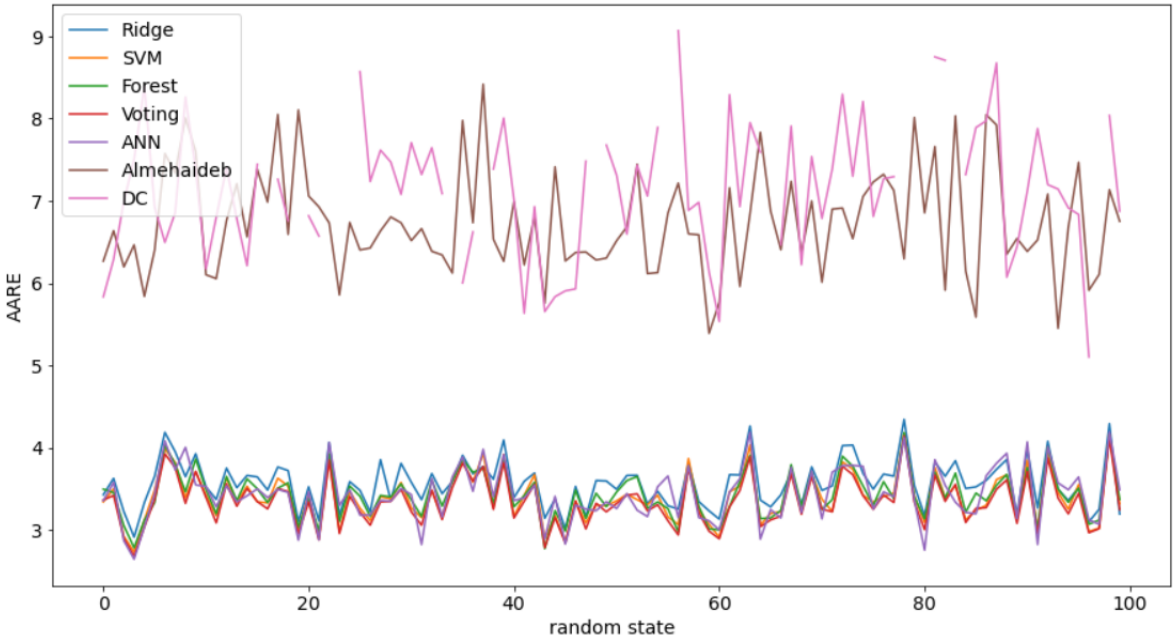


Figure 60: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

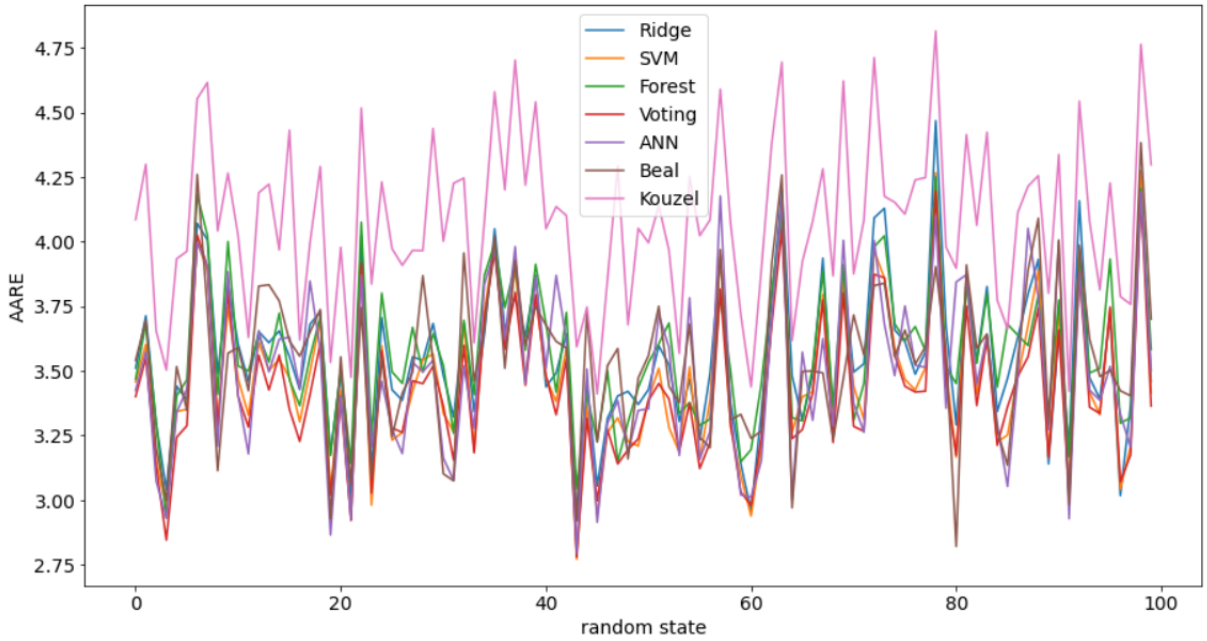


Figure 61: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

This extended run confirms our previous suggestion. Out of the 4 best literature

correlations, only Beal's performance is similar to ours, to the point where it can't be easily differentiated from ours. Some statistics will now be shown for those models.

	Ridge	SVM	Forest	Voting	ANN	Almehaideb	DC
count	100.00	100.00	100.00	100.00	100.00	100.00	87.00
mean	3.56	3.40	3.43	3.34	3.43	6.74	7.15
std	0.29	0.30	0.29	0.29	0.34	0.65	0.80
min	2.91	2.73	2.77	2.68	2.65	5.39	5.10
25%	3.36	3.18	3.24	3.14	3.21	6.30	6.77
50%	3.56	3.38	3.41	3.33	3.42	6.65	7.20
75%	3.70	3.58	3.60	3.50	3.64	7.13	7.60
max	4.34	4.18	4.18	4.14	4.19	8.42	9.07

Figure 62: $\mu = f(\mu_{ob}, \Delta P, P, P_b, GOR, API, T)$

	Ridge	SVM	Forest	Voting	ANN	Beal	Kouzel
count	100.00	100.00	100.00	100.00	100.00	100.00	100.00
mean	3.56	3.47	3.57	3.43	3.49	3.54	4.08
std	0.30	0.30	0.27	0.29	0.33	0.31	0.33
min	2.92	2.77	2.93	2.78	2.79	2.82	3.41
25%	3.38	3.26	3.40	3.24	3.27	3.34	3.87
50%	3.51	3.43	3.52	3.39	3.48	3.55	4.08
75%	3.69	3.65	3.73	3.59	3.70	3.73	4.26
max	4.47	4.27	4.25	4.20	4.19	4.38	4.81

Figure 63: $\mu = f(\mu_{ob}, \Delta P, P, P_b)$

Our best regressor (voting) is on average a bit over 0.1% better than Beal's, and the improvement gets closer to 0.2% if the extra features are included. Out of the one hundred instances that we run, the voting regressor was a better predictor 75% of the time and this percentage goes up to 82% if the extra features are included. On average, every one of our regressors except ridge and forest (for the simple inputs) are better than Beal's. This result is very important. The regressors were basically trained with the same set of hyperparameters for different fluids one hundred times. The results were always similar, and almost always better than every literature correlation.

6 Conclusions

This chapter concludes our analysis on machine learning methods for the prediction of undersaturated oil viscosity. Our task of developing correlations that work better than literature's using machine learning techniques was a success. Despite not always being the clear winner, machine learning models were very easy to introduce and develop.

It was very interesting to see how easily could machine learning algorithms be developed. Even without feature engineering, random forest regression could train a model with an AAE of just short of 4%. What this means is that even without understanding the problem at all, simply by inserting a few features and a target, a working model could be predicted in just minutes. Beal and the others probably spent months studying their data and working on creating a correlation.

Instead of going through that process, a decision tree could be trained and then by just using a boosting or bagging method, a great predictor would be created in mere minutes. In this work the majority of the popular regression methods were examined, all of them provided acceptable results and the process of developing was very easy thanks to Python's high level libraries that perform the complex algorithms mentioned in the fourth chapter by simply running a few lines of code. Not only that, but this is a machine learning application in a real life problem that worked in practice and not just in theory. One can only imagine how much better results could have been given a dataset ten times bigger than this one. No wonder algorithms had on average a lower AAE when μ_{ob} was less than 5 cp. Besides the more predictable nature of the fluid's viscosity change in this condition, the abundance of data points was certainly a major factor for that.

The final code for this thesis can be found on "https://github.com/flammmes/mythesis".

A Statistical error analysis

A.1 Literature correlations

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Beal (1946)	-0.92	5.07	3.00	4.18
Kouzel (1965)	1.46	6.84	4.23	5.57
Vazquez and Beggs (1976)	6.18	13.33	8.11	12.25
Khan (1987)	3.26	7.70	5.18	6.57
Petrosky (1990)	5.35	26.24	6.72	25.92
Kartoatmodjo and Schmidt (1991)	-7.23	7.16	7.27	7.11
Orbey and Sandler (1993)	-0.80	5.75	3.31	4.77
Kouzel Modified (1997)	-2.76	6.08	3.93	5.40
Hossain (2005)	3.35	7.21	5.42	5.81

$$0 \leq \mu_{ob} \leq 1$$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Beal (1946)	-3.69	5.60	4.38	5.08
Kouzel (1965)	2.66	6.81	4.53	5.73
Vazquez and Beggs (1976)	12.61	20.49	13.42	19.97
Khan (1987)	1.59	6.35	4.07	5.12
Petrosky (1990)	2.72	8.09	5.71	6.33
Kartoatmodjo and Schmidt (1991)	-5.19	6.07	5.54	5.75
Orbey and Sandler (1993)	-2.73	5.57	3.95	4.79
Kouzel Modified (1997)	-0.32	5.88	3.78	4.51
Hossain (2005)	6.11	7.61	6.92	6.88

$$1 \leq \mu_{ob} \leq 5$$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Beal (1946)	-2.84	5.83	3.93	5.16
Kouzel (1965)	2.62	7.35	5.45	5.58
Vazquez and Beggs (1976)	12.71	22.83	14.16	21.96
Khan (1987)	-2.77	6.43	4.80	5.10
Petrosky (1990)	-15.60	10.87	15.60	10.87
Kartoatmodjo and Schmidt (1991)	-1.69	5.87	3.99	4.63
Orbey and Sandler (1993)	-7.34	6.90	7.66	6.55
Kouzel Modified (1997)	0.89	6.62	4.69	4.75
Hossain (2005)	6.25	7.90	7.45	6.77

$$5 \leq \mu_{ob} \leq 20$$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Beal (1946)	0.56	7.65	4.82	5.97
Kouzel (1965)	-0.66	7.91	5.64	5.58
Vazquez and Beggs (1976)	8.08	20.47	13.20	17.61
Khan (1987)	-11.66	8.28	11.67	8.27
Petrosky (1990)	-26.27	14.49	26.27	14.49
Kartoatmodjo and Schmidt (1991)	1.40	7.61	4.94	5.95
Orbey and Sandler (1993)	-16.19	9.91	16.19	9.91
Kouzel Modified (1997)	-1.30	7.73	5.65	5.43
Hossain (2005)	1.21	7.48	5.41	5.32

$$20 \leq \mu_{ob} \leq 50$$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Labedi Libya (1982)	-5.66	5.90	5.89	5.67
Labedi Nigeria (1982)	-6.11	6.59	6.63	6.07
Al-Khafaji (1987)	-0.23	11.02	4.82	9.91
Abdul-Majeed (1990)	1.98	14.92	6.67	13.49
De Ghetto (1994)	-5.56	5.58	5.56	5.57
Almehaideb (1997)	4.47	11.73	6.41	10.79
Elsharkawy (1999)	-6.66	5.86	6.71	5.80
Dindoruk and Christman (2001)	2.14	5.78	4.31	4.41

$$0 \leq \mu_{ob} \leq 1$$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Labedi Libya (1982)	-7.22	7.36	7.65	6.91
Labedi Nigeria (1982)	-7.41	9.44	9.17	7.75
Al-Khafaji (1987)	-7.06	7.41	7.51	6.95
Abdul-Majeed (1990)	47.53	240.79	50.67	240.15
De Ghetto (1994)	-7.46	8.69	8.57	7.60
Almehaideb (1997)	2.39	9.89	6.52	7.81
Elsharkawy (1999)	-9.98	7.89	9.98	7.89
Dindoruk and Christman (2001)	5.39	7.04	6.40	6.14

$$1 \leq \mu_{ob} \leq 5$$

	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Labedi Libya (1982)	-12.71	10.69	13.45	9.75
Labedi Nigeria (1982)	-14.79	9.98	14.89	9.82
Al-Khafaji (1987)	-14.20	10.52	14.50	10.09
Abdul-Majeed (1990)	37.60	270.20	46.88	268.75
De Ghetto (1994)	-4.01	11.49	8.34	8.87
Almehaideb (1997)	-1.71	8.93	6.17	6.67
Elsharkawy (1999)	-16.31	10.78	16.35	10.72
Dindoruk and Christman (2001)	10.89	9.41	11.37	8.83

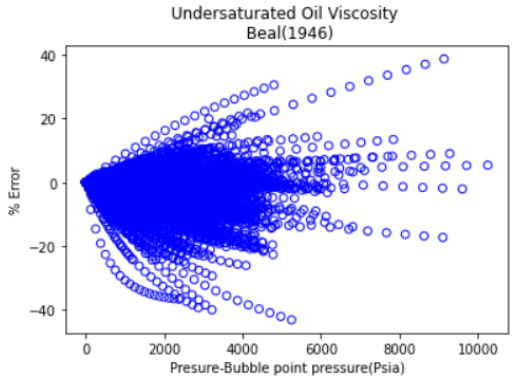
$$5 \leq \mu_{ob} \leq 20$$

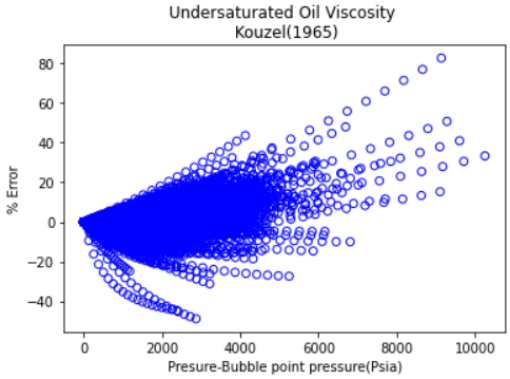
	% AE	% Std Dev AE	%AAE	% Std Dev AAE
Labedi Libya (1982)	-23.83	13.09	23.83	13.09
Labedi Nigeria (1982)	-24.93	14.12	24.93	14.12
Al-Khafaji (1987)	-25.30	13.69	25.30	13.69
Abdul-Majeed (1990)	54.85	304.34	85.31	297.25
De Ghetto (1994)	-2.80	7.41	5.93	5.25
Almehaideb (1997)	-7.59	14.60	13.01	10.06
Elsharkawy (1999)	-26.26	14.28	26.26	14.28
Dindoruk and Christman (2001)	50.88	38.22	50.88	38.22

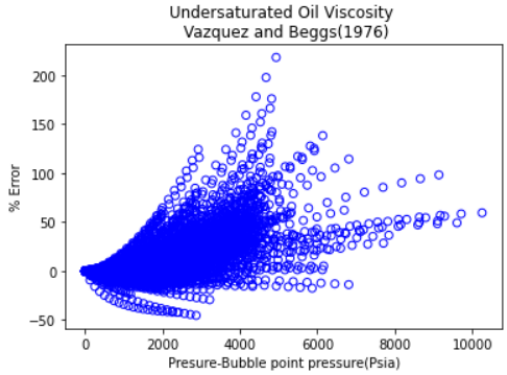
$$20 \leq \mu_{ob} \leq 50$$

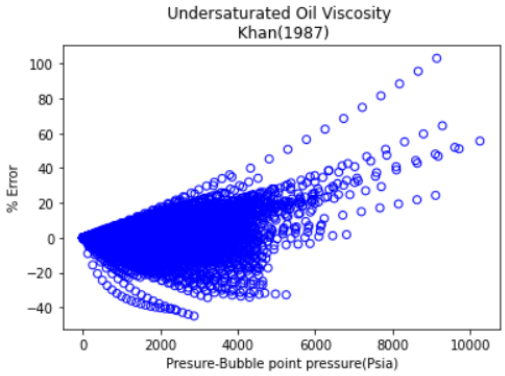
B Graphical error analysis

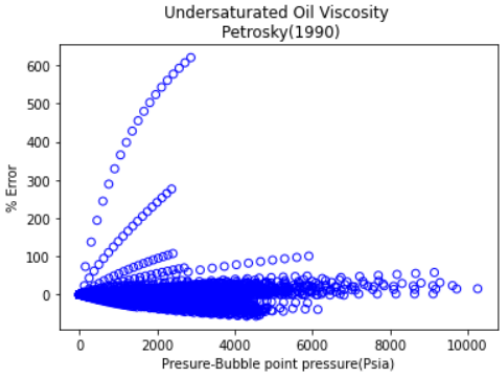
B.1 Literature correlations

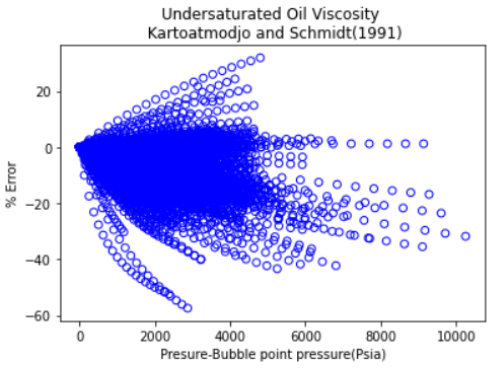


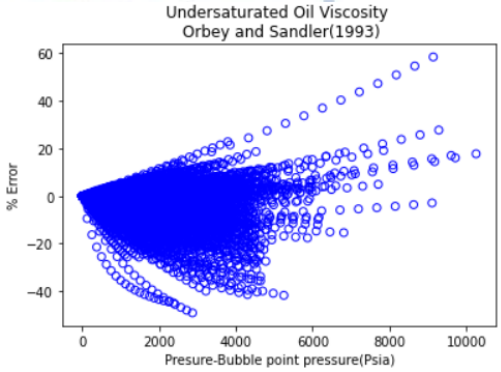


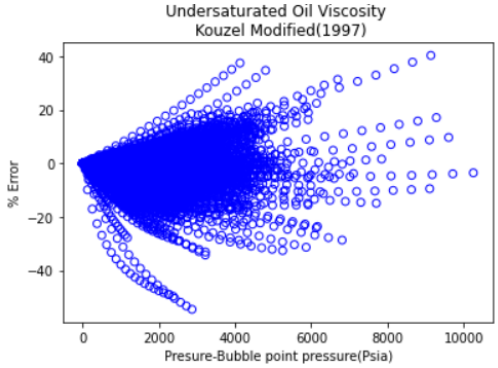


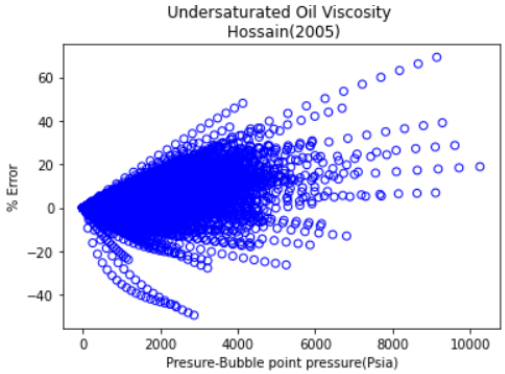


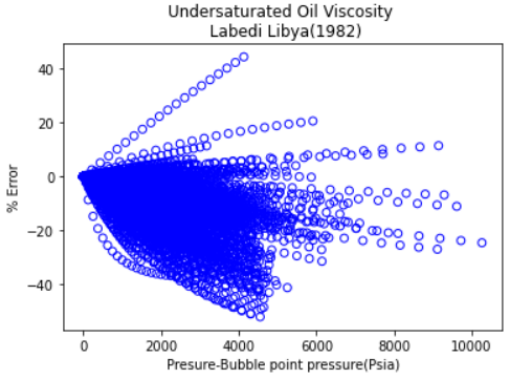


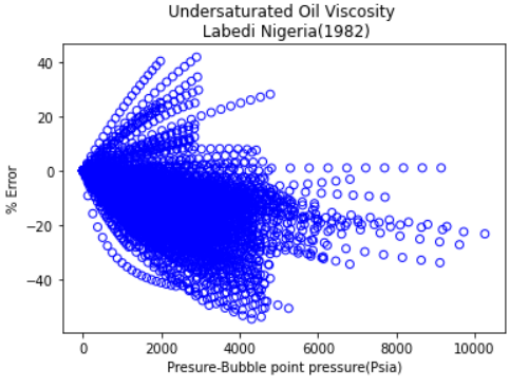


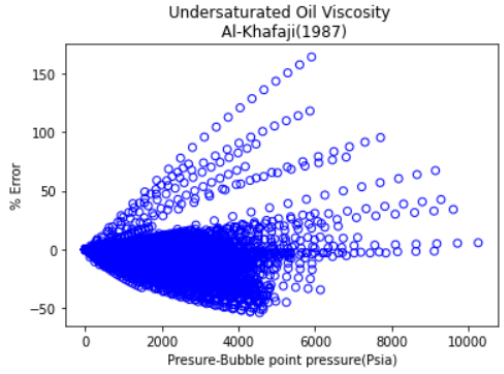


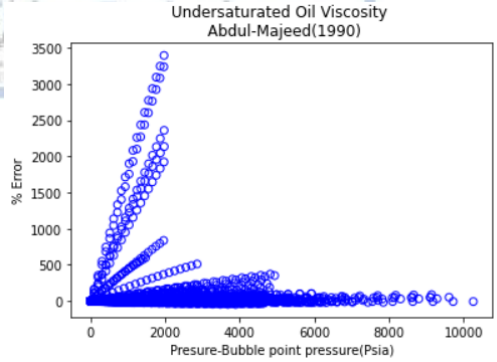


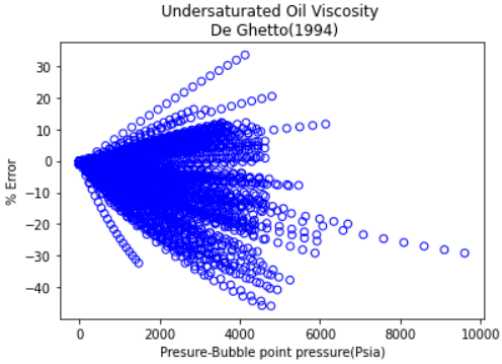


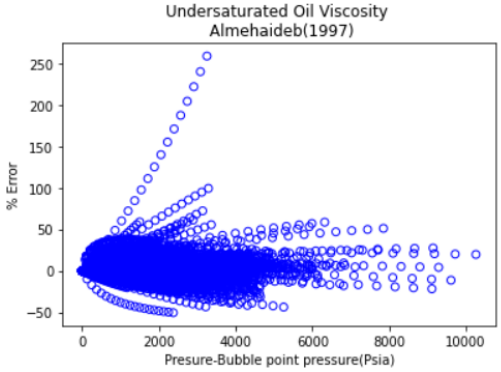


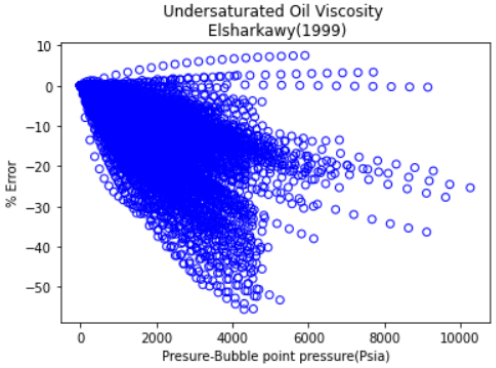


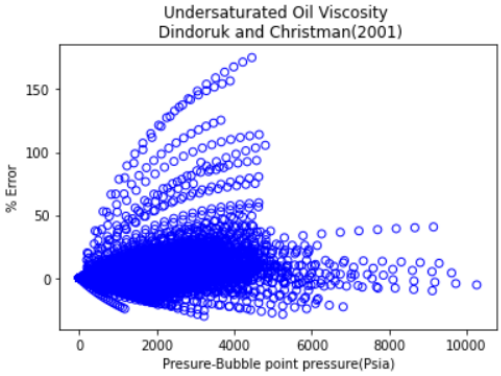


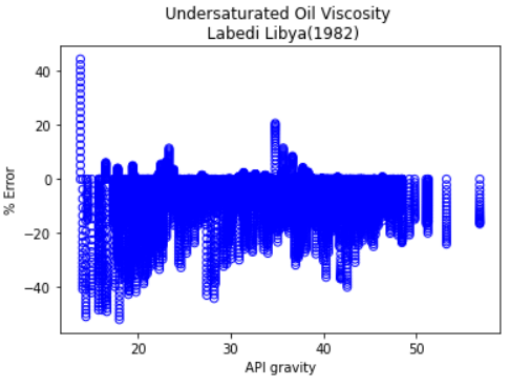


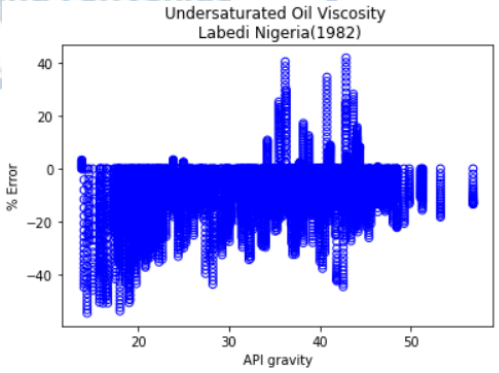


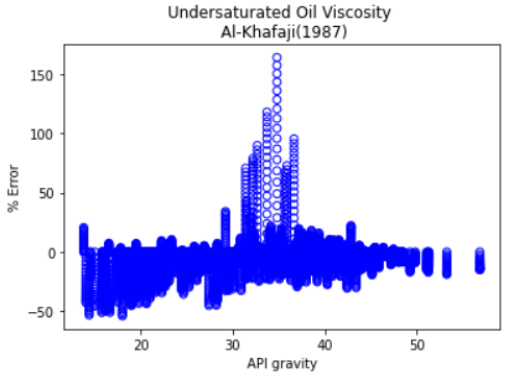


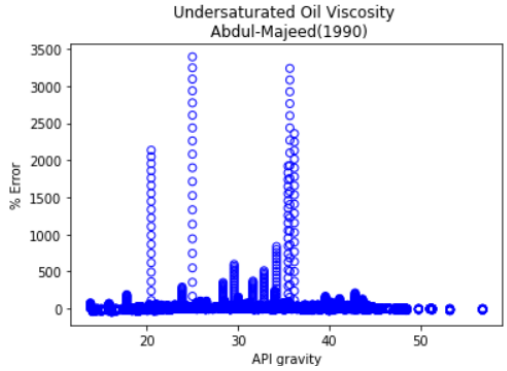


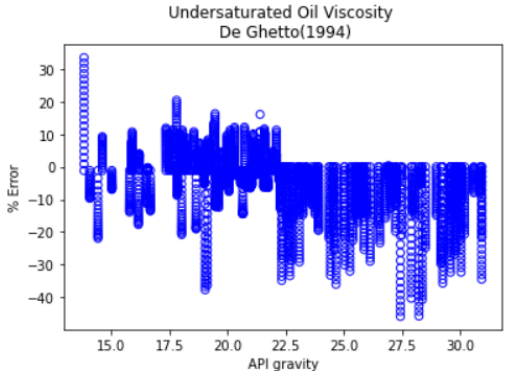


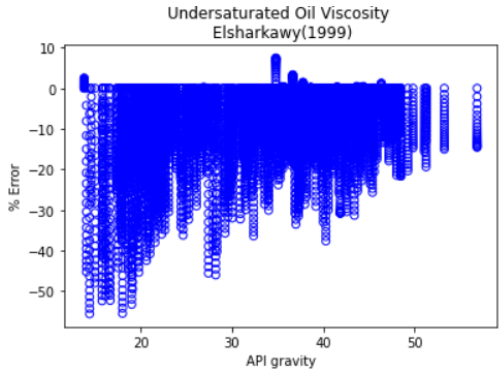


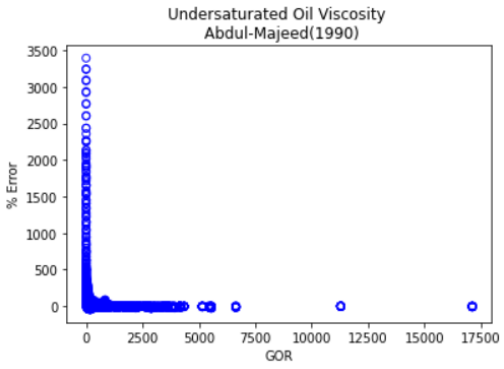


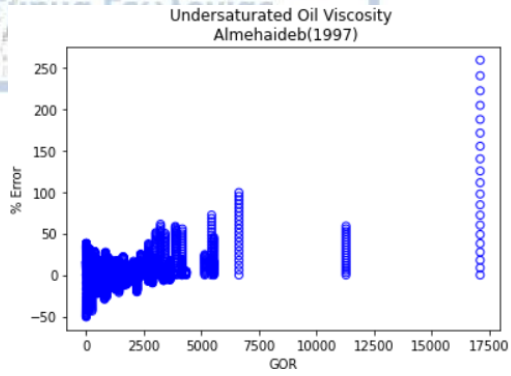


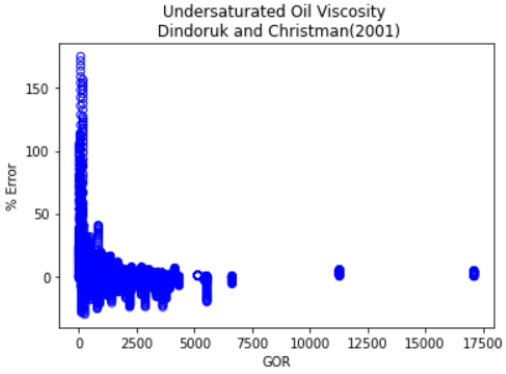


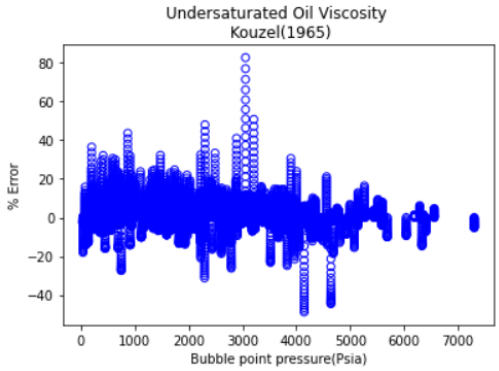


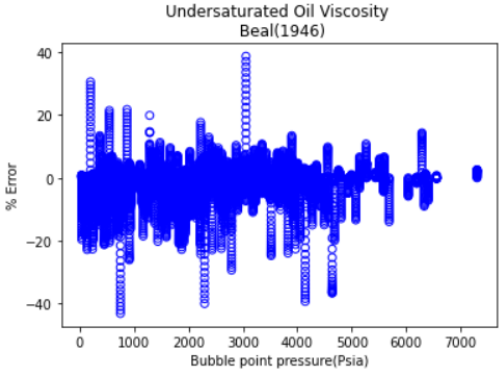


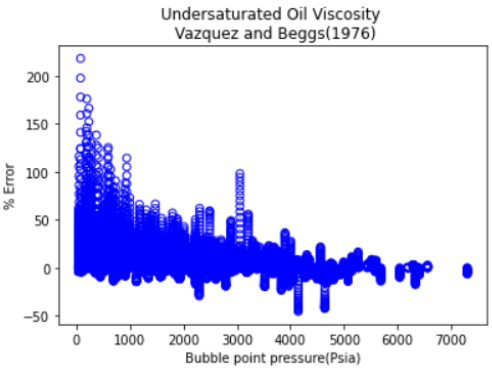


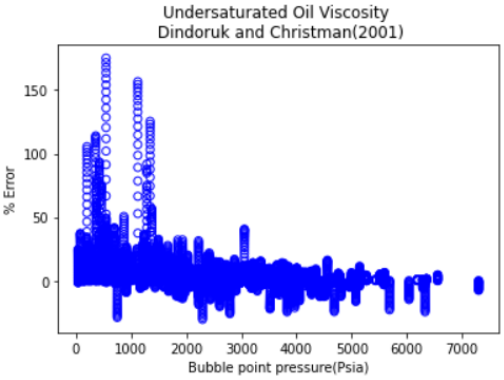


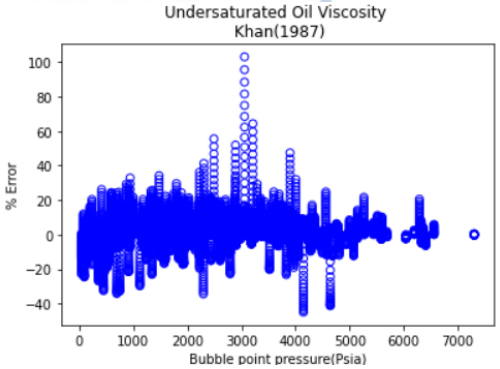


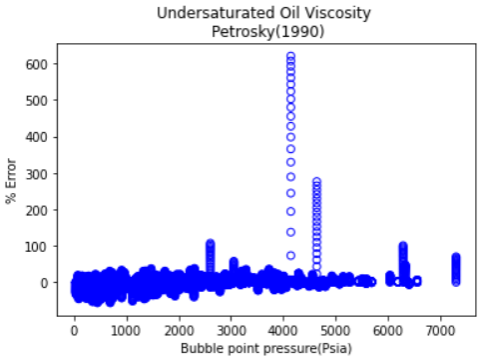


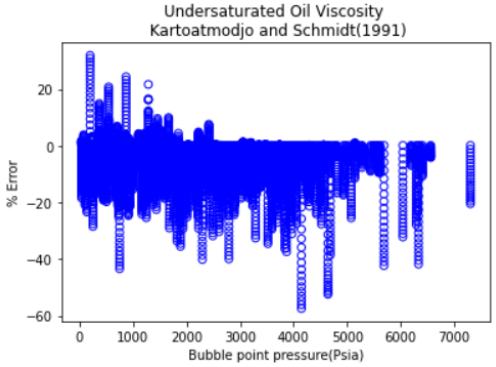


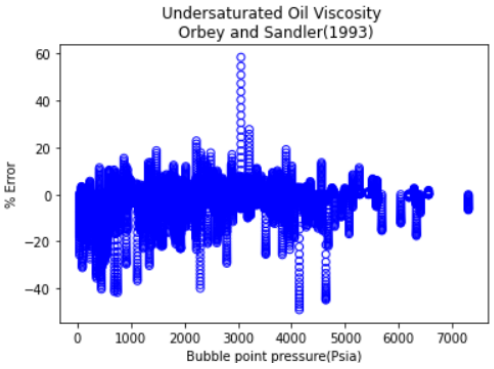


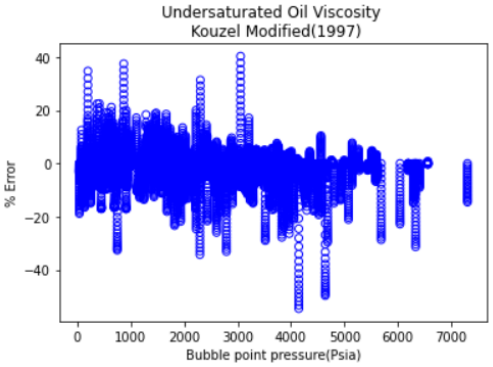


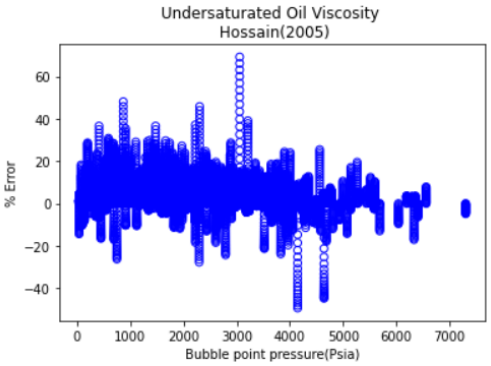


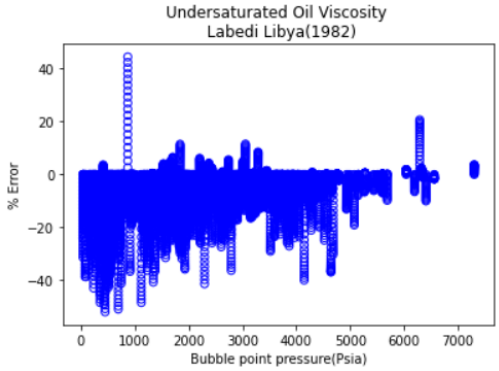


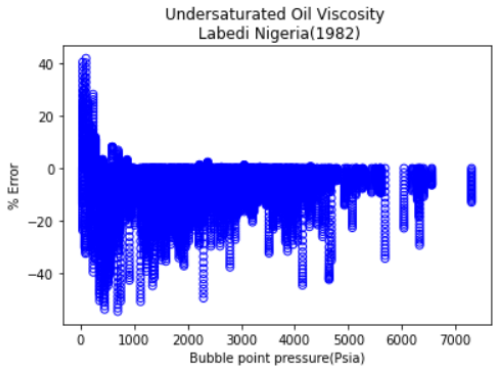


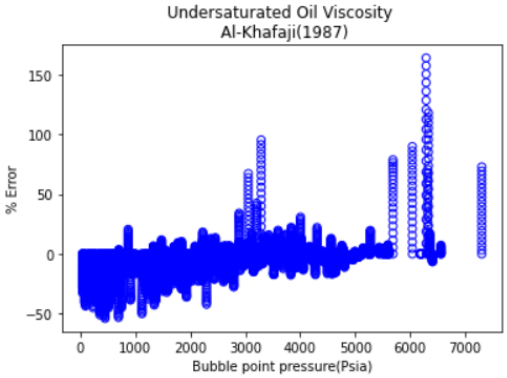


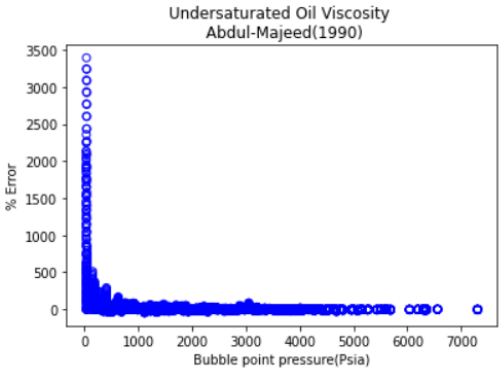


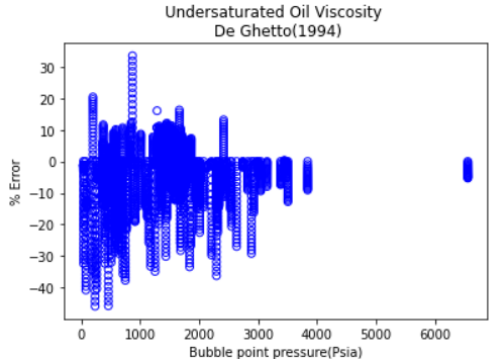


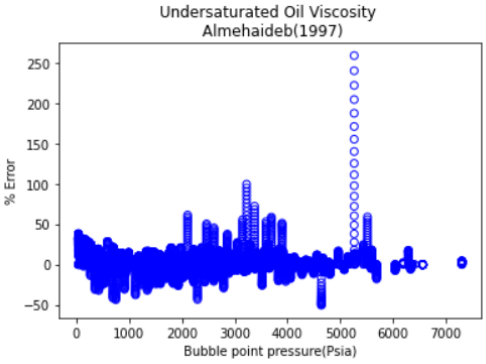


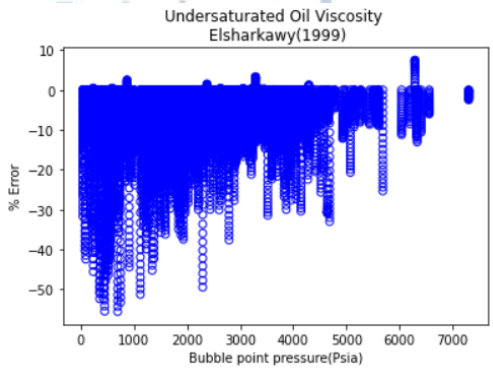


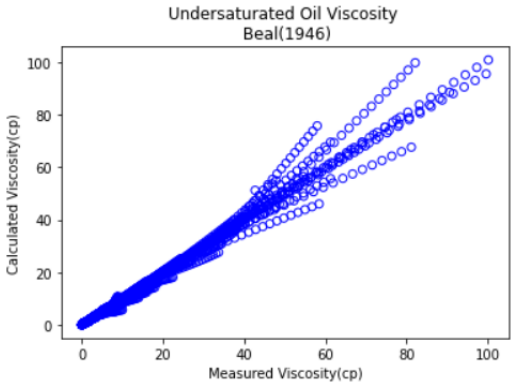


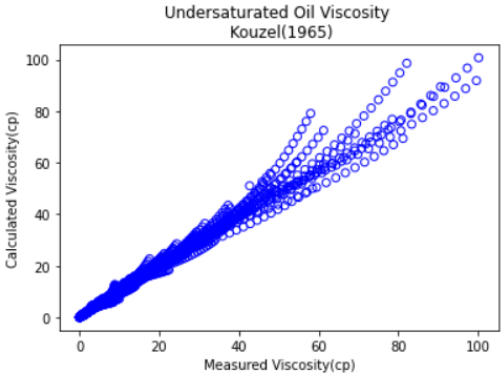


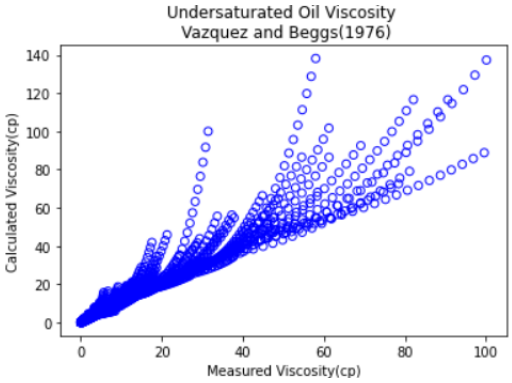


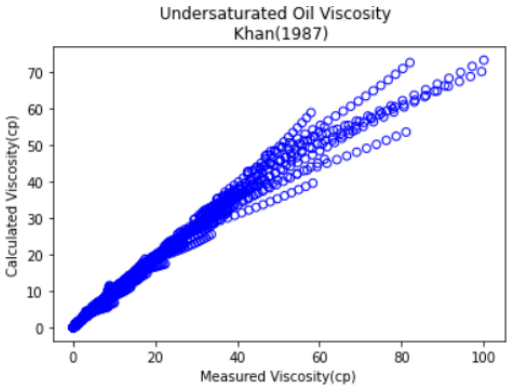


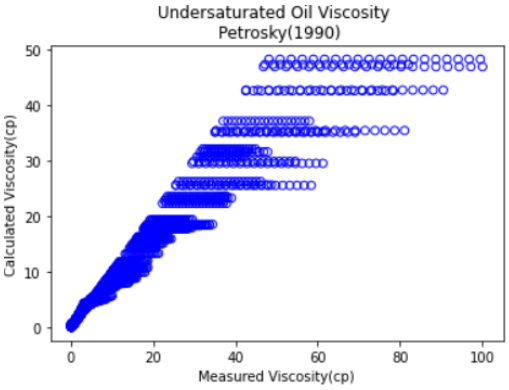


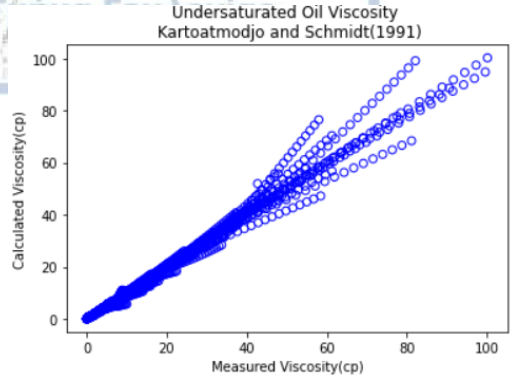


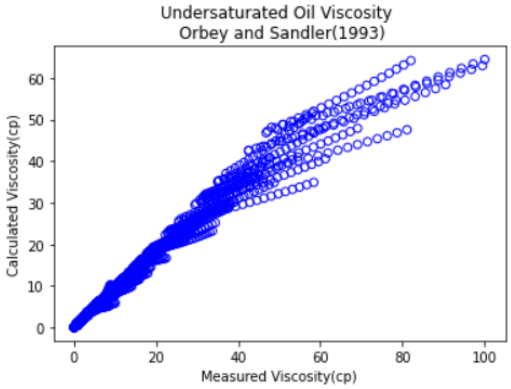


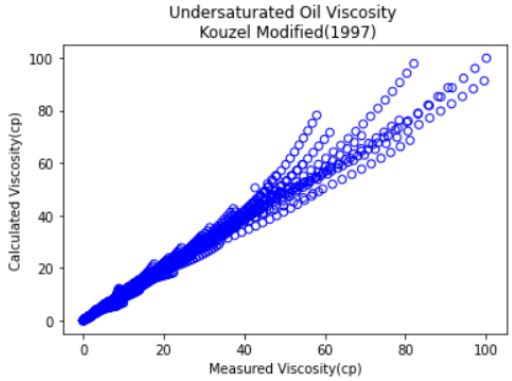


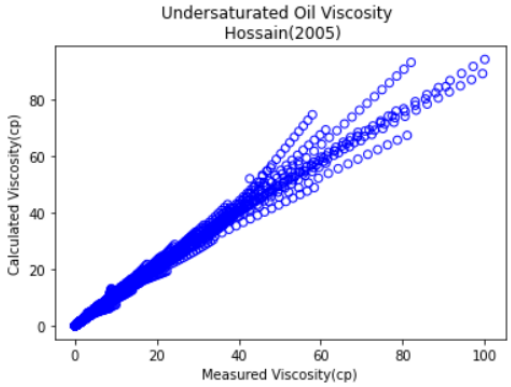


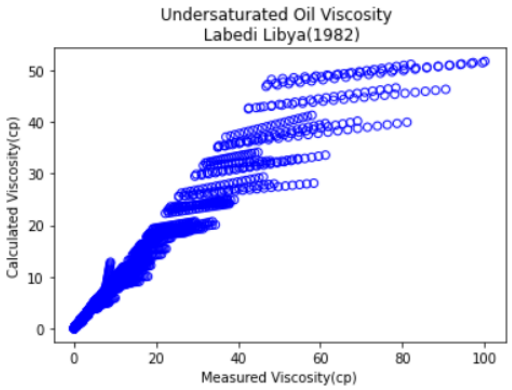


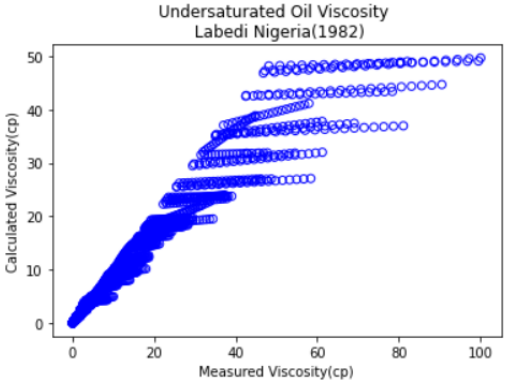


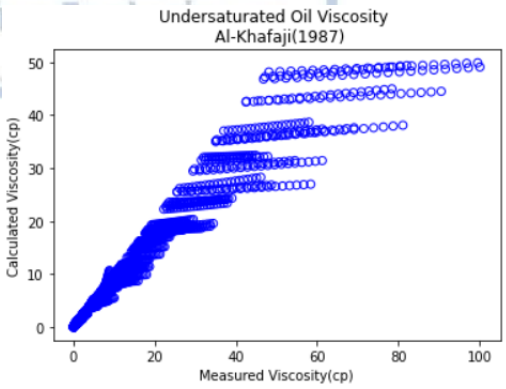


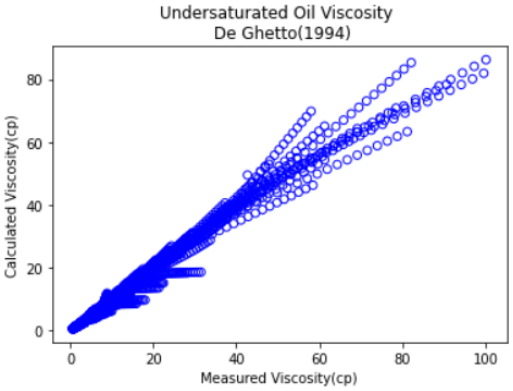


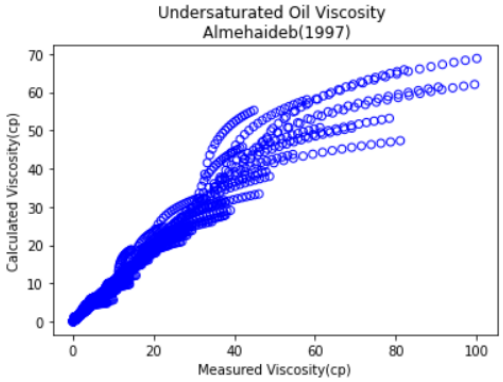


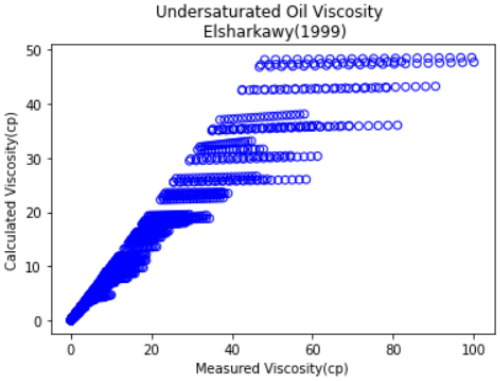


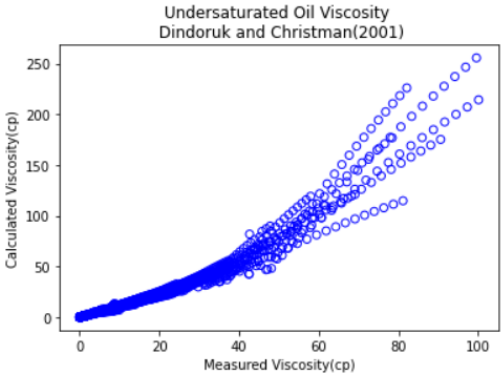


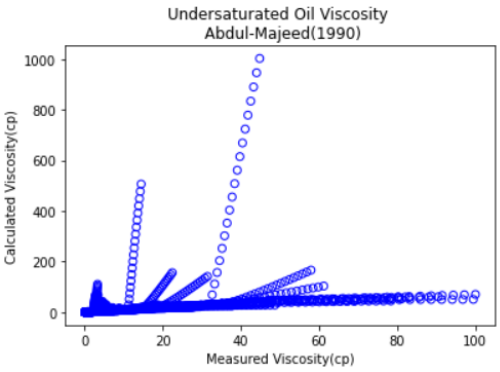












References

Abdul-Majeed, G. H., Kattan, R. R., & Salman, N. H. (1990). New correlation for estimating the viscosity of undersaturated crude oils. *Journal of Canadian Petroleum Technology*, *29*(03).

Al-Khafaji, A. H., Abdul-Majeed, G. H., Hassoon, S. F., et al. (1987). Viscosity correlation for dead, live and undersaturated crude oils. *J. Pet. Res*, *6*(2), 1–16.

Almehaideb, R. (1997). Improved pvt correlations for uae crude oils. In *Middle east oil show and conference.*

API. (1997). Technical data book—petroleum refining. *Metric Edition, Vols*, *1*(2).

Beal, C. (1946). The viscosity of air, water, natural gas, crude oil and its associated gases at oil field temperatures and pressures. *Transactions of the AIME*, *165*(01), 94–115.

Bergman, D., & Sutton, R. P. (2006). Undersaturated oil viscosity correlation for adverse conditions. In *Spe annual technical conference and exhibition.*

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Cristianini, N., Shawe-Taylor, J., et al. (2000). *An introduction to support vector machines and other kernel-based learning methods.* Cambridge university press.

De Ghetto, G., & Villa, M. (1994). Reliability analysis on pvt correlations. In *European petroleum conference.*

Dindoruk, B., & Christman, P. G. (2004). Pvt properties and viscosity correlations for gulf of mexico oils. *SPE Reservoir Evaluation & Engineering*, *7*(06), 427–437.

Elsharkawy, A., & Alikhan, A. (1999). Models for predicting the viscosity of middle east crude oils. *Fuel*, *78*(8), 891–903.

Fletcher, T. (2009). Support vector machines explained. *Tutorial paper*, 1–19.

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems.* " O'Reilly Media, Inc.".

Glaso, O. (1980). Generalized pressure-volume-temperature correlations. *Journal of Petroleum Technology*, *32*(05), 785–795.

Hossain, M. S., Sarica, C., Zhang, H.-Q., Rhyne, L., & Greenhill, K. (2005). Assessment and development of heavy oil viscosity correlations. In *Spe international thermal operations and heavy oil symposium.*

Introduction to gp¶. (n.d.). Retrieved from `https://gplearn.readthedocs.io/en/stable/intro.html`

Kartoatmodjo, T., Schmidt, Z., et al. (1991). New correlations for crude oil physical properties. *paper SPE*, *23556*.

Khan, S., Al-Marhoun, M., Duffuaa, S., & Abu-Khamsin, S. (1987). Viscosity correlations for saudi arabian crude oils. In *Middle east oil show.*

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kouzel, B. (1965). How pressure affects liquid viscosity. *Hydrocarbon Processing.*

March, *1965*, 120.

Kulchanyavivat, S. (2005). *The effective approach for predicting viscosity of saturated and undersaturated reservoir oil*. Texas A&M University.

Labedi, R. M. (1982). Pvt correlations of the african crudes. *1980-1989-Mines Theses & Dissertations*.

Mukherjee, H., & Brill, J. P. (1999). Multiphase flow in wells. *Society of Petroleum Engineers of AIME*.

Orbey, H., & Sandler, S. I. (1993). The prediction of the viscosity of liquid hydrocarbons and their mixtures as a function of temperature and pressure. *The Canadian Journal of Chemical Engineering*, *71*(3), 437–446.

Petrosky, G. E. (1990). *Pvt correlations for gulf of mexico crude oils* (Unpublished doctoral dissertation). University of Southwestern Louisiana.

Rojas, R. (1996). The backpropagation algorithm. In *Neural networks* (pp. 149–182). Springer.

Shawe-Taylor, J., Cristianini, N., et al. (2004). *Kernel methods for pattern analysis*. Cambridge university press.

Standing, M., & Katz, D. (1981). Volumetric and phase behavior of oil hydrocarbon system. *Society of Petroleum Engineers of AIME, Dallas*.

Standing, M. B. (1977). *Volumetric and phase behavior of oil field hydrocarbon systems*. Society of petroleum engineers of AIME.

Vazquez, M., & Beggs, H. (1980). *Correlations for fluid physical property prediction. jpt 32 (6): 968–970* (Tech. Rep.). SPE-6719-PA. DOI: 10.2118/6719-PA.