Data in Brief 32 (2020) 106201

Contents lists available at ScienceDirect

Data in Brief

journal homepage: www.elsevier.com/locate/dib

Data Article

Data on assessment and exploratory statistical correlation data analysis of sintered Nd:YAG laser welded 2507 duplex stainless steel

Ayorinde Tayo Olanipekun[a,*], Peter Madindwa Mashinini[a], Nthabiseng Beauty Maledi[b]

[a] Department of Mechanical and Industrial Engineering Technology, University of Johannesburg, South Africa
[b] School of Chemical and Metallurgical Engineering, University of the Witwatersrand, Johannesburg, South Africa

ARTICLE INFO

Article history:
Received 30 June 2020
Revised 8 August 2020
Accepted 14 August 2020
Available online 20 August 2020

Keywords:
Spark plasma sintering
Welding
Microhardness
Data analysis

ABSTRACT

The effect of Spark Plasma Sintering (SPS) parameters and Nd:YAG laser welding parameters such as sintering temperature, sintering time, welding power, welding speed, on Vickers hardness mechanical properties of the Weld Zone(WZ) was statistically investigated. Rectangular plates of 2507 Duplex Stainless Steel (DSS) fabricated by SPS were joined by Nd: YAG laser welding process. Linear regression analysis was performed on the data, followed by Pearson correlation analysis to relate hardness with other predicting variables and show their statistical significance to hardness value of the weld zone.The evaluated R^2 which is 49% was used to measure the closeness of the data to the regression fitted line.

* **Corresponding author**.
E-mail address: atolanipekun@uj.ac.za (A.T. Olanipekun).

https://doi.org/10.1016/j.dib.2020.106201

Specifications Table

Subject	Mechanical Engineering and Materials science
Specific subject area	Data analysis, Welding Engineering and Nanotechnology
Type of data	Table Chart Graph Figure
How data were acquired	The data was acquired using Future-Tech Microhardness tester (FM-700) Software: Google Tensor flow, Python and Jupyter
Data format	Raw analyzed Filtered Normalized
Parameters for data collection	• Sintering process: 2507 DSS powder samples was sintered at the following temperatures (S-temp) 900 °C, 1000 °C, 1100 °C at a pressure of 50 MPa under vaccum. A heating rate of 100 °C/m was considered for the sintering process that lasted for between (S-time) 5min-10 min. • Nd:YAG laser welding: The laser power and welding speed adopted in this research for Sample A, B and C were Laser power (Wel-power) of 1500 W, 1700 W, 2000 W, while the welding speed employed are 3 m/min, 3 m/min and 2 m/min respectively • Vickers Hardness test: Vickers microhardness (HV) test was carried out using Vickers microhardness (Future Tech FM-700) tester, applying a load of 300 gf with dwell time of 10 s at room temperature.
Description of data collection	The data used for the stastical analysis was gotten from three experimnetal processes, Sintering parameters and hardness test data. The welding parameters were selected based on the optimized parameters from literature. The Vickers microhardness test data was targeted at the welded zone (WZ) for this research.
Data source location	University of Witwatersrand, Johannesburg, South Africa
Data accessibility	With the article Mendeley Data https://data.mendeley.com/datasets/c49p4g34cc/2#folder-795deb94-da97–479a-b3ec-71154d68a4bd

Value of the Data

- The data is beneficial to researchers that may likely want to work in the area of data and machine learning analysis of engineering materials.
- Data utilized as a reference to investigate the statistical coefficient of sintered and welded data on the overall mechanical properties of 2507 DSS and regression analysis.
- Further research on microhardness analysis of welded specimen can be built on the Vickers analysis carried out in this research.

1. Data Description

In this information dispensation there is enormous data like feedbacks, medical data, materials data and share data e.t.c., data science has in no doubt assisted us to make quality decisions in all sphere of life. Recently, material science and engineering field has accepted data science as a tool to analyse materials properties data. Data science has help to greatly reduce the cost save time in material structure design, overall reduce time in material design and its behaviour [1]. The data and analyses included here corroborate the statistical analysis drawn from the study of ND:YAG laser welded 2507DSS. Chemical composition data analysis of the powdered 2507 (DSS) Table 1. Data sets in Table 2. Summarizes the experimental data gotten from the sintering process and laser welding of 2507 DSS which includes the sintering temperature, sintering time,

Table 1
Chemical composition of the As-received powdered 2507 DSS (wt%).

Material	C	P	Si	Ni	N	Mn	S	Cr	Mo	Fe
2507	0.02	0.014	0.4	7.1	0.31	0.8	0.001	24.4	3.76	Balance

Table 2
Sintering and welding parameters [3].

S-temp (°C)	S-time (min)	Wel-power (W)	Wel-speed(m/min)	Hardness (HV)
1000	5	1500	3	343.7
1000	5	1500	3	350.1
1000	5	1500	3	351.7
1000	5	1500	3	365.6
1000	5	1500	3	337.5
1000	5	1500	3	362.8
1000	5	1500	3	356.7
1000	5	1500	3	357
1000	5	1500	3	356.9
1000	5	1500	3	361.2
900	10	1700	3	347.6
900	10	1700	3	325
900	10	1700	3	310
900	10	1700	3	331.8
900	10	1700	3	349.9
900	10	1700	3	339.3
900	10	1700	3	386.1
900	10	1700	3	347.9
900	10	1700	3	381.8
900	10	1700	3	355.7
1100	5	2000	2	354
1100	5	2000	2	379
1100	5	2000	2	380.4
1100	5	2000	2	373.7
1100	5	2000	2	380.9
1100	5	2000	2	401.4
1100	5	2000	2	396.9
1100	5	2000	2	392.7
1100	5	2000	2	377.2
1100	5	2000	2	395.1

Table 3
Pearson correlation between Hardness and S-time.

	Hardness (HV)	S-time (min)
Hardness(HV)	1.0000	−0.4539
S-time (min)	−0.4539	1.0000

welding power, welding speed and hardness. Fig. 1 shows the graphics detail seaborn plot for all the data. Fig 2. Shows the heat mapping representation for the data. Fig. 3 depicts the linear regression plot for predicted value and true value of the hardness. Fig. 4 represents the linear regression plot for sintering time(S-time) against hardness. Fig. 5 represents the linear regression plot for welding power (Wel-power) against hardness. Fig. 6 describes the linear regression plot for welding speed (Wel-speed) against hardness. Fig. 7 illustrates the linear regression plot for sintering temp (S-temp) against hardness.

Table 3 –Table 6 explain the extent of pearson correlation of hardness to the predictor values.

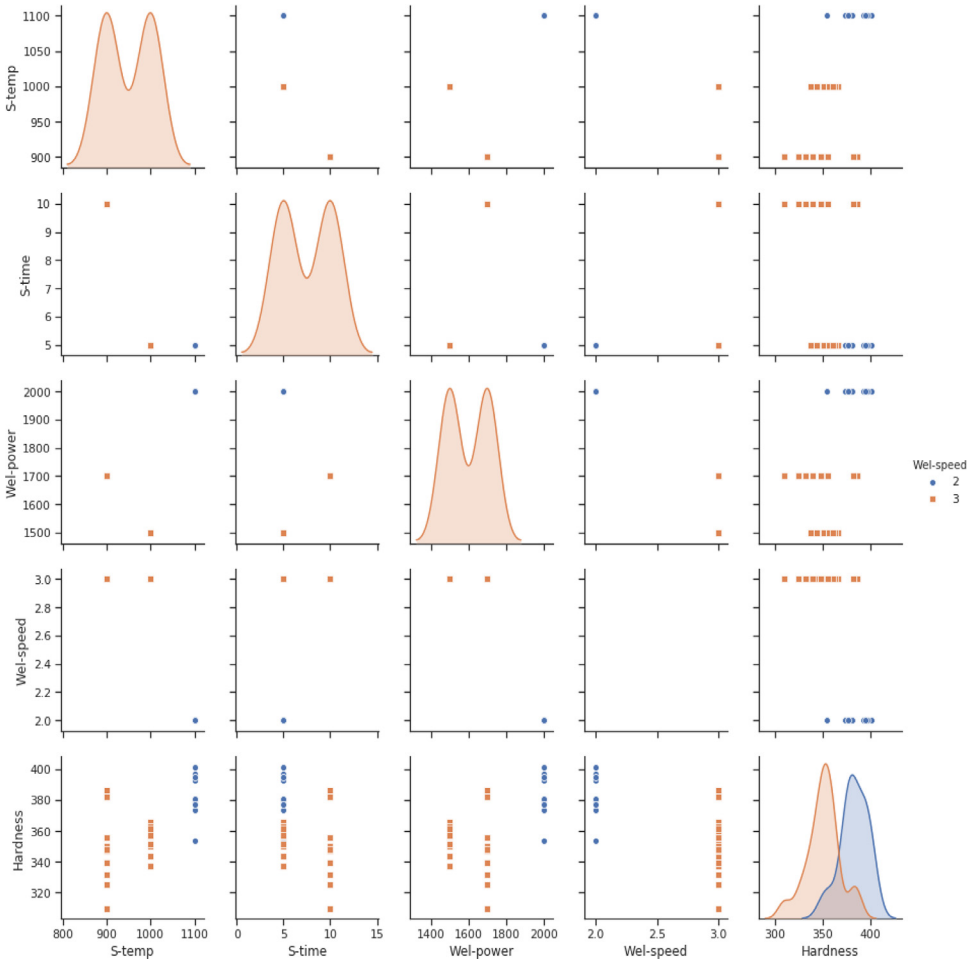

Fig. 1. Seaborn plot.

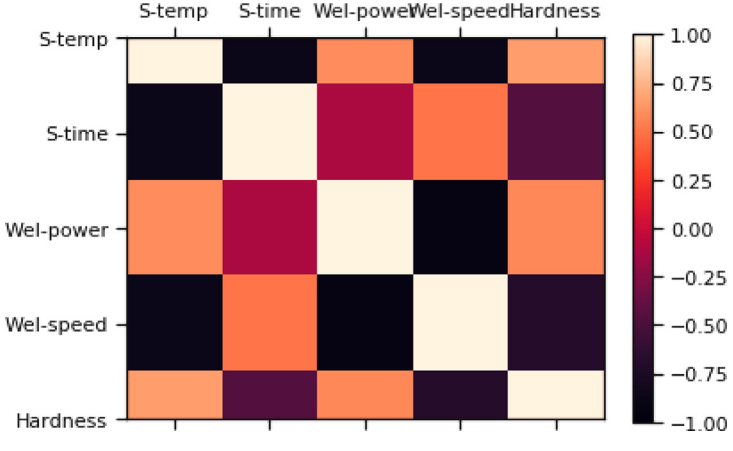

Fig. 2. Heat map visualization.

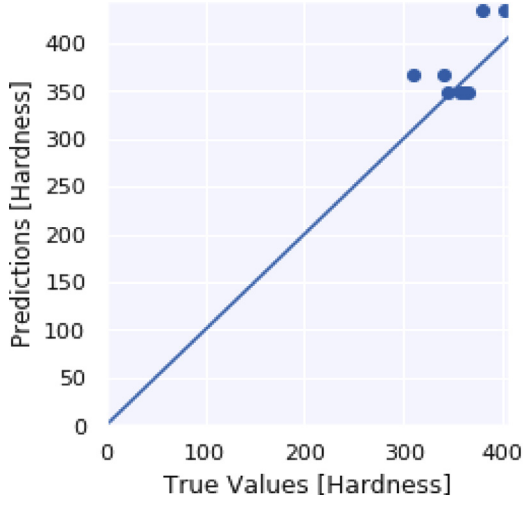

Fig. 3. Linear regression plot of hardness prediction values against the true value.

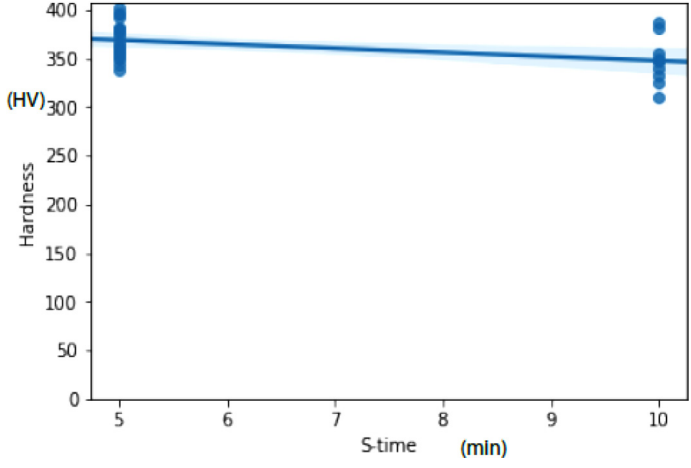

Fig. 4. Regression plot of hardness against Sintering time.

Table 4
Pearson correlation between Hardness and Wel-power.

	Hardness (HV)	Wel-power (W)
Hardness(HV)	1.0000	0.5824
Wel-power(W)	0.5824	1.0000

Table 5
Pearson correlation between Hardness and Wel-speed.

	Hardness (HV)	Wel-speed (m/min)
Hardness(HV)	1.0000	−0.6892
Wel-speed(m/min)	−0.6892	1.0000

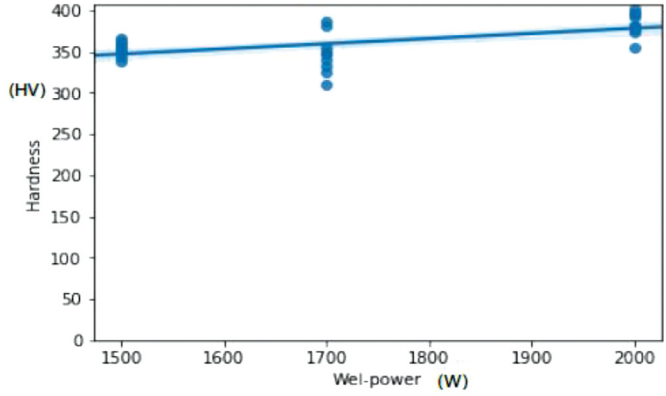

Fig. 5. Regression plot of hardness against welding power.

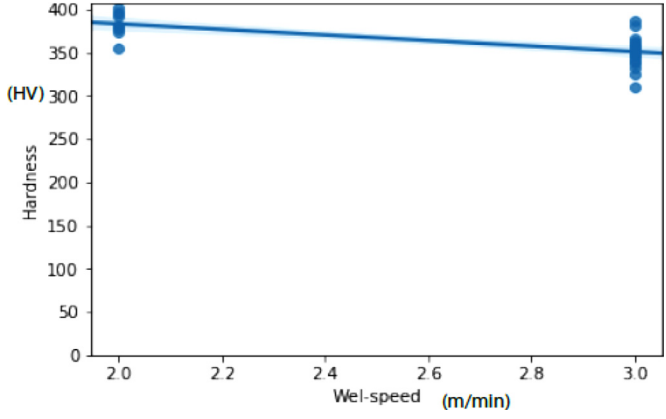

Fig. 6. Pearson correlation of hardness and welding speed.

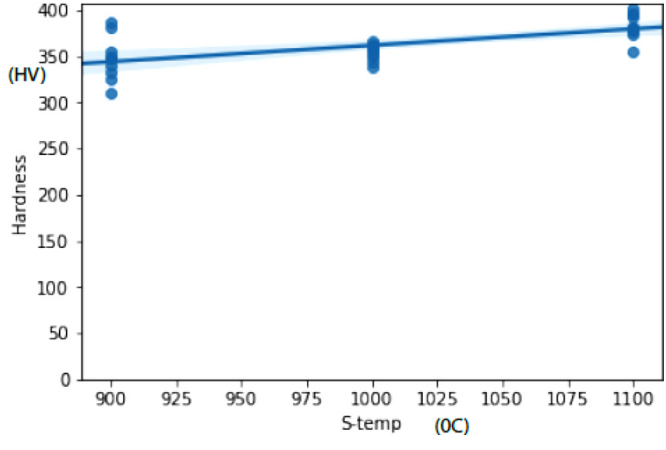

Fig. 7. Regression plot of hardness against sintering temperature.

Table 6
Pearson correlation between Hardness and S-temp.

	Hardness(HV)	S-temp (°C)
Hardness	1.0000	−0.6892
S-temp (°C)	0.6600	1.0000

2. Experimental design, materials, and methods

2.1. Fabrication of the materials using sintering and welding processes

2507 DSS powdered material whose compositions is shown in Table 1, was sintered with spark plasma sintering equipment SPS equipment (model HHPD-25, FCT system GmbH, Rauenstein, Germany, situated at Tshwane university of Technology Pretoria, South Africa). Meanwhile, the welding process proceded with the cutting of the sintered 2507 two parts of size (size: 12 × 12 × 3 mm^3), which was later prepared in butt configuration for the welding process. Laser welding equipment used is JK 600 pulsed Nd:YAG laser welding machine, at the council for scientific and industrial research (CSIR) South Africa. Also, parameters used in the data analysis was gotten from the sintering and welding processes as show in Table 2. Which form our predictor variables.

2.2. Microhardness

Hardenss of welded samples was carried using Vickers microhardness (Future Tech FM-700) tester, applying a load of 300 gf with dwell time of 10 s at room temperature. In this research, we consider Vickers microhardness at the weld zone (WZ). It has been indicated in literature that the chemical composition of the WZ goes along way to detrermine mechanical integrity of the welded metallic alloy, and it is also worthy to note that the microstructure of the WZ differs from the base metal (BM) because of the thermal history variation and differnce in chemical composition [2]. Also, for the data analsis, the Microhadness data form our targeted variable.

2.3. Data analysis

A raw dataset comprising of 30 instances was subjected to preprocesing using sklearn packges in tensor flow, for data cleaning, normalization, followed by splitting the data into 25% testing and 75%training set and eventually subjected it to regression analysis and visualized using different packages in tensor flow. The determination coefficient R^2, Pearson correlation coefficient, were determination coefficient on which we based benchmarking and prediction, respectively. Prediction data-set was used to test the model, also, process taking for the data analysis is shown in the taxonomical figure below.

Meanwhile, linear model was imported from scikit-learn, and from sklearn.linear_model, linear regression is imported, we then define the predictor variable and target variable. Normalization was performed using the Eq. (1) below.

$$Y = \frac{(Ymax - Ymin) * (X - Xmin)}{Xmax - Xmin} + Ymin \tag{1}$$

Where Y are the normalized datasets and X are input datasets. Eq. (2) was used to calculate the output datasets.

Eq. (2) represents the simple linear progression,

$$y = b_0 + b_1 x \tag{2}$$

x represent the predictor (independent) variable while target (dependent) variable y, while b_0 is intercept and b_1 is slope.

$$\text{Coefficient of determination or } R^2 = \left(1 - \frac{MSE\ of\ regression\ line}{MSE\ of\ the\ average\ data}\right) \tag{3}$$

MSE is mean square error, and R^2 is a measure or meteric we have useed to determine how close the dats is to fitted regression line in Fig. 3 to Fig. 7.

Pearson correlation is another metric we used to measure the strength of correlation between the hardness values and other predictor values.

2.3.1. Data visualization

Seaborn plot showed in Fig. 1 shows stastitical info-graphics four the data, providing high-level interface for stastical analysis and variation of hardness to dependent variables 'wel-power', 'S-temp', 'S-time', and 'wel-speed'.

Heat map plots in Fig. 2 was used to visualize data and it shows target variable (Hardness) proportional to colour with respect to variables, 'wel-power', 'S-temp', 'S-time', and 'wel-speed' in the vertical and horizontal axis respectively. This allows us to visualize the Hardness is related to 'wel-power', 'S-temp', 'S-time', and 'wel-speed'.

2.4. Code implemented

"""Copy of Copy of Copy of Copy of model.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/github/AyorindeTayo/Artificial-Neural-Network-for-Mechanical-Hardness-property-of-Welded-DSS-/blob/master/Copy_of_Copy_of_Copy_of_Copy_of_model.ipynb⟨?PMU?⟩

In this notebook, I am going to implement a simple linear regression model with tensorflow. We are going to predict the Hardness of a welded material in term of the welding power, speed, time.

"""

```
# importing the tensorflow package and other auxilary packages
import tensorflow as tf
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from tensorflow import keras
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler,PolynomialFeatures
from sklearn.datasets import make_classification
from matplotlib import pyplot as plt
from sklearn.linear_model import LogisticRegression
sns.set()
```

"""##1. Loading the data with pandas"""

```
# Loading the data with the pandas read_csv attribute
from google.colab import files
```

```
files.upload()

df=pd.read_excel('data_1004.xlsx')
data = pd.DataFrame(df)

def boxplot(df,x,y):
  ax = sns.boxplot(x="x", y="x", data=df)

# Renaming the feature and looking at the data
data

# The shape of our data.
data.shape

# looking at the type of the data
data.dtypes

"""##1. Box plotting"""
ax = sns.boxplot(x="S-temp", y="Hardness", data=df)

ax = sns.boxplot(x="S-time", y="Hardness", data=df)

ax = sns.boxplot(x="Wel-power", y="Hardness", data=df)

ax = sns.boxplot(x="Wel-speed", y="Hardness", data=df)

df.plot(kind='box',figsize=(15,15), subplots=True, layout=(3,3), sharex=False, sharey=False)
plt.show()

"""One could do more plotting but we are ok now to build up our model. But first of all let us write a function that will help normalize our data.

##3. Definig the function to normalize the data
Normalization is very important in machine learning as building a model on a raw data set may result in poor performance of the model. It is always advisable to do so before feeding the data into your machine learning algorithm.
"""

#This function will return the normalized data
def Normalize(x):
  return (x-np.mean(x))/np.std(x)

"""##4- Splitting the data in test and train set"""

#I steal this from tensorflow tutorial. One could also used the split function in sklearn.
train_set=data.sample(frac=0.75,random_state=0)
test_set=data.drop(train_set.index)

"""## 5. Data visualisation and statistics"""

sns.set() sns.pairplot(train_set, height=3);

train_set.describe()

"""## 6. Defining the labels"""

train_labels=train_set.pop('Hardness')
test_labels=test_set.pop('Hardness')

"""##3. Defining and compiling the model
Next we will create the simplest possible neural network. It has 1 layer, and that layer has 1 neuron, and the input shape to it is just 1 value.
```

```
"""

def build_model():
  model = keras.Sequential([
  keras.layers.Dense(64, activation=tf.nn.relu, input_shape=[len(train_set.keys())]),
  keras.layers.Dense(64, activation=tf.nn.relu),
  keras.layers.Dense(1)
])

optimizer = tf.keras.optimizers.RMSprop(0.001)

model.compile(loss='mse',
    optimizer=optimizer,
    metrics=['mae','mse'])
return model

model=build_model()
model.summary()

"""##
Training the model
Here we are going to train our model by feeding into the model the training sets of data (features and labels).
"""

history = model.fit(
  train_set, train_labels,
  epochs=150, validation_split = 0.2, verbose=0)

hist = pd.DataFrame(history.history)
hist['epoch'] = history.epoch
hist.tail()

"""## Plotting the results"""

def plot_history(history):
  hist = pd.DataFrame(history.history)
  hist['epoch'] = history.epoch

plt.figure()
plt.xlabel('Epoch')
plt.ylabel('MAE for concrete strength')
plt.plot(hist['epoch'], hist['mean_absolute_error'],
  label='Train
  Error') plt.plot(hist['epoch'], hist['val_mean_absolute_error'],
  label = 'Val Error')
#plt.ylim([0,5])
plt.legend()

plt.figure()
plt.xlabel('Epoch')
plt.ylabel('MSE [concrete strength]')
plt.plot(hist['epoch'], hist['mean_squared_error'],
  label='Train Error')
plt.plot(hist['epoch'], hist['val_mean_squared_error'],
  label = 'Val Error')
#plt.ylim([0,20])
plt.legend()
```

```
plt.show()

plot_history(history)

model=build_model()
early_stop = keras.callbacks.EarlyStopping(monitor='val_loss', patience=10)

history = model.fit(train_set, train_labels, epochs=150,
        validation_split = 0.2, verbose=0, callbacks=[early_stop])

plot_history(history)

hist = pd.DataFrame(history.history)
hist['epoch'] = history.epoch
hist.tail()

loss, mae, mse = model.evaluate(test_set, test_labels, verbose=0)

print("Testing set Mean Abs Error: {:5.2f} Hardness".format(mae))

test_predictions = model.predict(test_set).flatten()

plt.scatter(test_labels, test_predictions)
plt.xlabel('True Values [Hardness]')
plt.ylabel('Predictions [Hardness]')
plt.axis('equal')
plt.axis('square')
plt.xlim([0,plt.xlim()[1]])
plt.ylim([0,plt.ylim()[1]])
_ = plt.plot([-1000, 1000], [-1000, 1000])

test_predictions

test_labels

Normalize(train_set)

Normalize(train_labels)

sns.pairplot(data, hue="Hardness", palette="husl")

"""Plotting *italicized text*"""

df.plot(kind='density',figsize=(15,15), subplots=True, layout=(3,3), sharex=False)
plt.show()

import seaborn as sns; sns.set(style="ticks", color_codes=True)
sns.pairplot(data)

sns.pairplot(data, hue="Hardness")

sns.pairplot(data, hue="Wel-speed", markers=["o", "s"])

sns.pairplot(data, kind="reg")

corr_matrix=df.corr()
corr_matrix

names=['S-temp','S-time','Wel-power','Wel-speed','Hardness']
fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(corr_matrix, vmin=-1, vmax=1)
fig.colorbar(cax)
```

```
ticks = np.arange(0,5,1)
ax.set_xticks(ticks)
ax.set_yticks(ticks)
ax.set_xticklabels(names)
ax.set_yticklabels(names)
plt.show()

"""**ANOVA ANALYSIS**"""

df.corr()

"""We can use the Pandas method corr() to find the feature other than price that is most correlated wit
price"""
df.corr()['Hardness'].sort_values()

sns.regplot(x="Hardness", y="Wel-speed", data=df)
plt.ylim(0,)

df[["Hardness", "Wel-speed"]].corr()

sns.regplot(x="Hardness", y="Wel-power", data=df)

plt.ylim(0,)

df[["Hardness", "Wel-power"]].corr()

sns.regplot(x="S-temp", y="Hardness", data=df)
plt.ylim(0,)

df[["Hardness", "S-temp"]].corr()

sns.regplot(x="Hardness", y="S-time", data=df)
plt.ylim(0,)

df[["Hardness", "S-time"]].corr()

from scipy import stats

pearson_coef, p_value = stats.pearsonr(df['Hardness'], df['Wel-power'])
print("The pearson Correlation Coefficient is", pearson_coef, "with a p-value of p =", p_value)

pearson_coef, p_value = stats.pearsonr(df['Hardness'], df['S-temp'])
print("The pearson Correlation Coefficient is", pearson_coef, "with a p-value of p =", p_value)

pearson_coef, p_value = stats.pearsonr(df['Hardness'], df['S-time'])
print("The pearson Correlation Coefficient is", pearson_coef, "with a p-value of p =", p_value)

pearson_coef, p_value = stats.pearsonr(df['Hardness'], df['Wel-speed'])
print("The pearson Correlation Coefficient is", pearson_coef, "with a p-value of p =", p_value)

"""To see if different types 'Wel-power' impact 'Hardness"

**MODEL DEVELOPMENT**
"""

import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

X = df[['Wel-power']]
Y = df['Hardness']
lm = LinearRegression()
lm.fit(X,Y)
```

```
lm.score(X, Y)

features =["S-temp", "S-time","Wel-power","Wel-speed"]

"""We can Fit a linear regression model using the longitude feature 'long' and caculate the R^2."""

X=df[['S-temp','S-time','Wel-power','Wel-speed']]
Y=df['Hardness']
lm.fit(X,Y)
lm.score(X,Y)

"""Create a list of tuples, the first element in the tuple contains the name of the estimator:'scale'
'polynomial' 'model'"""

Input=[('scale',StandardScaler()),('polynomial',
PolynomialFeatures(include_bias=False)),('model',LinearRegression())]

"""We use the list to create a pipeline object, predict the 'price', fit the object using the features in the list features, then fit the model and calculate the R^2"""
Input= [('scale', StandardScaler()), ('polynomial',
PolynomialFeatures(include_bias=False)),('model',LinearRegression())]
pipe=Pipeline(Input)
pipe
X=df[['S-temp','S-time','Wel-power','Wel-speed']]
Y=df['Hardness']

pipe.fit(X,Y)

pipe.score(X,Y)

"""**MODEL EVALUATION AND REFINEMENT**"""

from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
print("done")

we will split the data into training and testing set""" features =["S-temp", "S-time","Wel-power","Wel-speed"]

X = df[features]
Y = df['Hardness']

x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.15, random_state=1)

print("number of test samples:", x_test.shape[0])
print("number of training samples:",x_train.shape[0])

Create and fit a Ridge regression object using the training data, setting the regularization parameter to 0.1 and calculate the R^2 using the test data.**"""

from sklearn.linear_model import Ridge

RigeModel = Ridge(alpha=0.1)
RigeModel.fit(x_train, y_train)
yhat = RigeModel.predict(x_test)
Rsqu_test= RigeModel.score (x_test,y_test)
print(Rsqu_test)
```

```
"""**Perform a second order polynomial transform on both the training data and testing data. Create and fit a Ridge regression object using the training data, setting the regularisation parameter to 0.1. Calculate the R^2**"""

pr=PolynomialFeatures(degree=2)
x_train_pr= pr.fit_transform(x_train[['S-temp', 'S-time','Wel-power','Wel-speed']])
x_test_pr= pr.fit_transform(x_test[['S-temp', 'S-time','Wel-power','Wel-speed']])
RigeModel.fit(x_train_pr, y_train)
yhat = RigeModel.predict(x_test_pr)
Rsqu_test= RigeModel.score (x_test_pr,y_test)
print(Rsqu_test)

"""**Artificial Neural Network Analysis**"""

input_vector = np.array([2, 4, 11])
print(input_vector)

input_vector = np.array(input_vector, ndmin=2).T
print(input_vector, input_vector.shape)

import numpy as np
number_of_samples = 30
low = −1
high = 0
s = np.random.uniform(low, high, number_of_samples)
# all values of s are within the half open interval [−1, 0):
print(np.all(s >= −1) and np.all(s < 0))

plt.hist(s)
plt.show()

s = np.random.binomial(10, 0.5, 30)
plt.hist(s)
plt.show()

from scipy.stats import truncnorm
s = truncnorm(a=−2/3., b = 2/3., scale=1, loc=0).rvs(size=1000)
plt.hist(s)
plt.show()

def truncated_normal(mean=0, sd=1, low=0, upp=10):
  return truncnorm(
    (low - mean) / sd, (upp - mean) / sd, loc=mean, scale=sd)
X = truncated_normal(mean=0, sd=0.4, low=−0.5, upp=0.5)
s = X.rvs(10,000)
plt.hist(s)
plt.show()

X1 = truncated_normal(mean=2, sd=1, low=1, upp=10)
X2 = truncated_normal(mean=5.5, sd=1, low=1, upp=10)
X3 = truncated_normal(mean=8, sd=1, low=1, upp=10)
import matplotlib.pyplot as plt
fig, ax = plt.subplots(3, sharex=True)
ax[0].hist(X1.rvs(10000), normed=True)
ax[1].hist(X2.rvs(10000), normed=True)
ax[2].hist(X3.rvs(10000), normed=True)
plt.show()
```

```
no_of_input_nodes = 3
no_of_hidden_nodes = 4
rad = 1 / np.sqrt(no_of_input_nodes)
X = truncated_normal(mean=2, sd=1, low=-rad, upp=rad)
wih = X.rvs((no_of_hidden_nodes, no_of_input_nodes))
wih
```

```
no_of_hidden_nodes = 4
no_of_output_nodes = 2
rad = 1 / np.sqrt(no_of_hidden_nodes) # this is the input in this layer!
X = truncated_normal(mean=2, sd=1, low=-rad, upp=rad)
who = X.rvs((no_of_output_nodes, no_of_hidden_nodes))
```

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.dib.2020.106201.

References

[1] N. Sandhya, A survey on data science approach to predict mechanical properties of steel, in: International Conference on E-Business and Telecommunications, Springer, 2019, pp. 501–511.
[2] R. Gunn, Duplex Stainless Steels: Microstructure, Properties and Applications, Elsevier, 1997.
[3] Olanipekun Ayorinde; MashininiMadindwa; Maledi Nthabiseng “Data on sintering, Nd:YAG welding and vickers hardness value of 2507 duplex stainless steel (DSS))”, Mendeley Data, v2, (2020). 10.17632/c49p4g34cc.2