Supplementary Materials

Abbreviations, Theoretical Analysis Calculation

(*For reviewer's reference purpose and not included in the manuscript*)

1. **Coding for Machine Learning Modelling**
2. **Abbreviations**
3. **Theoretical Analysis Calculation**

```
# -*- coding: utf-8 -*-
"""Classifier_Paper_CEE.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1cLZ-pvge526T73QId-
O6Dl6udhJFKwFV
"""

#========================Import libraries
import numpy as np
import scipy.stats as stats

import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

import sklearn
from sklearn.ensemble import RandomForestClassifier
from sklearn.datasets import make_classification
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score
from sklearn import metrics

#========================== Load data
data =
pd.read_csv('/content/drive/MyDrive/Workshop/Aravind_CEE/ML_CEE_Final.csv
')
print(data.shape)
data.head()
data.tail()
data.isnull().sum()

#========================== extract required data
X =
data.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shea
r','Com_con_MPa','Eff_span_mm','Xsect_area_sqmm','TS_stir_Mpa','Xsect_sti
r','TS_MS_Mpa','TS_gfrp_Mpa']]
#'samp_desig',
X.head()
X['Shear_type'].unique()
y=data.loc[:,['Shear_F']] # 'Shear_F','Flexure_F',
'Crushing','Delamination'
y.head()


#========================== encoding
from sklearn.preprocessing import LabelEncoder
enc = LabelEncoder()
```

```
X.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shear']
] = \
X.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shear']
].apply(enc.fit_transform)
y.loc[:,['Shear_F']] = \
y.loc[:,['Shear_F']].apply(enc.fit_transform)

#X.loc[:,['samp_desig','Shear_type','str_50','GFRP_full','U_50','U_Full',
'Steel_Shear','Shear_F','Flexure_F','Crushing','Delamination']] = \
#X.loc[:,['samp_desig','Shear_type','str_50','GFRP_full','U_50','U_Full',
'Steel_Shear','Shear_F','Flexure_F','Crushing','Delamination']].apply(enc
.fit_transform)

X.head()
X['Shear_type'].unique()
y.head()

#========================== training and testing RF
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.3,random_state=0)

# evaluate random forest algorithm for classification
from numpy import mean
from numpy import std
from sklearn.datasets import make_classification
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import RepeatedStratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import ConfusionMatrixDisplay

#X_train, y_train = make_classification(n_samples=13000, n_features=13,
n_informative=10, n_redundant=3,random_state=0, shuffle=False)
model = RandomForestClassifier(n_estimators=4,max_depth=11)
#cv=RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=0)
#n_scores = cross_val_score(model, X_train, y_train, scoring='accuracy',
cv=cv, n_jobs=-1, error_score='raise')
#print('Accuracy: %.3f (%.3f)' % (mean(n_scores), std(n_scores)))
model.fit(X_train, y_train)
yhat = model.predict(X_test)

#======================== model performance
conf_mat = confusion_matrix(y_test, yhat)
#print(conf_mat)
# Visualize it as a heatmap
import seaborn
seaborn.heatmap(conf_mat)
plt.show()

# Plot non-normalized confusion matrix
titles_options = [
    ("Confusion matrix, without normalization", None),
    ("Normalized confusion matrix", "true"),
```

```
]
for title, normalize in titles_options:
    disp = ConfusionMatrixDisplay.from_estimator(
        model,
        X_test,
        y_test,
        #display_labels=class_names,
        cmap=plt.cm.Blues,
        normalize=normalize,
    )
    disp.ax_.set_title(title)

    print(title)
    print(disp.confusion_matrix)
# Print the precision and recall, among other metrics
    print(metrics.classification_report(y_test, yhat, digits=3))
plt.show()

from sklearn.metrics import mean_absolute_error, r2_score

y_predict = model.predict(X_train)
mae= round(mean_absolute_error(y_train,y_predict),4)
rmse = round((np.sqrt(metrics.mean_squared_error(y_train, y_predict))),4)
r2 = round(r2_score(y_train,y_predict),2)

print("The model performance for training set")
print("--------------------------------------")
print('MAE is {}'.format(mae))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
print("\n")

y_predict = model.predict(X_test)
mae= round(mean_absolute_error(y_test,y_predict),4)
rmse = round((np.sqrt(metrics.mean_squared_error(y_test, y_predict))),4)
r2 = round(r2_score(y_test,y_predict),2)

print("The model performance for testing set")
print("--------------------------------------")
print('MAE is {}'.format(mae))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
print("\n")

# Calculate the absolute errors
errors = abs(y_predict - y_test)
# Print out the mean absolute error (mae)
print('Mean Absolute Error:', round(np.mean(errors), 2), 'kN.')

# Calculate mean absolute percentage error (MAPE)
mape = 100 * (errors / y_test)
# Calculate and display accuracy
accuracy = 100 - np.mean(mape)
print('Accuracy:', round(accuracy, 2), '%.')
```

```
import seaborn as sns
plt.figure(figsize=(5, 7))


ax = sns.distplot(y, hist=False, color='r', label='Actual Value')
sns.distplot(y_predict, hist=False, color='b', label='Fitted Values' ,
ax=ax)

plt.title('Actual vs Fitted Values')
ax.set(xlabel='Failure Load (kN)', ylabel='Density')
#ax.set(xlabel='Mid Span Deflection (in mm)', ylabel='Density')

plt.legend()
plt.show()
plt.close()

#========================== custom test
#Sl. 1
#model.predict([[3, 0, 0, 0, 0, 1, 31.21, 650.0, 13250, 250.0, 39.72,
454, 0]])
#sl.11,12 - phd work
#model.predict([[3, 0, 0, 0, 0, 1, 17.98, 1003.4, 13236, 420.8, 57.06,
415, 0]])
#model.predict([[3, 0, 0, 0, 0, 1, 17.98, 1003.4, 13236, 420.8, 57.06,
415, 0]])


#model.predict([[3, 0, 0, 0, 1, 0, 31.21, 2650.0, 13250, 210.0, 100, 500,
165]])
#model.predict([[3, 0, 0, 0, 0, 1, 29.8, 900, 13000, 325, 56.55, 420,
0]])
model.predict([[2, 0, 0, 0, 0, 0, 29.8, 900, 13000, 0, 0, 420, 0]])
#model.predict([[1, 0, 0, 0, 1, 0, 29.8, 900, 13000, 0, 600, 420, 420]])
#model.predict([[1, 0, 0, 0, 1, 0, 29.8, 900, 13000, 0, 1200, 420, 311]])
```

```
# -*- coding: utf-8 -*-
"""Paper_CEE.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1_--5-
x9t_D5MDitdSsAZc1R93aXS5ZgB
"""
#========================Import libraries
#from google.colab import drive
#drive.mount('/content/drive')
import numpy as np
import scipy.stats as stats
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

import sklearn
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score
from sklearn import metrics

#========================== Load data
data = pd.read_csv('/content/drive/MyDrive/Data/ML_CEE_Final.csv')
print(data.shape)
data.head()
data.tail()
data.isnull().sum()

#========================== extract required data
X =
data.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shea
r','Com_con_MPa','Eff_span_mm','Xsect_area_sqmm','TS_stir_Mpa','Xsect_sti
r','TS_MS_Mpa','TS_gfrp_Mpa']]
X.head()
X['Shear_type'].unique()
y=data.loc[:,['Mid_span_D_mm']] # Failure_Load_kN

#========================== encoding
from sklearn.preprocessing import LabelEncoder
enc = LabelEncoder()
X.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shear']
] = \
X.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shear']
].apply(enc.fit_transform)
X.head()
X['Shear_type'].unique()

#========================== training and testing RF
```

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.7,random_state=0)

from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators=4, max_depth=11)
model.fit(X_train,y_train)

from sklearn.metrics import mean_absolute_error, r2_score

y_predict = model.predict(X_train)
mae= round(mean_absolute_error(y_train,y_predict),4)
rmse = round((np.sqrt(metrics.mean_squared_error(y_train, y_predict))),4)
r2 = round(r2_score(y_train,y_predict),2)

#======================== model performance
print("The model performance for training set")
print("--------------------------------------")
print('MAE is {}'.format(mae))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
print("\n")

y_predict = model.predict(X_test)
mae= round(mean_absolute_error(y_test,y_predict),4)
rmse = round((np.sqrt(metrics.mean_squared_error(y_test, y_predict))),4)
r2 = round(r2_score(y_test,y_predict),2)

print("The model performance for testing set")
print("--------------------------------------")
print('MAE is {}'.format(mae))
print('RMSE is {}'.format(rmse))
print('R2 score is {}'.format(r2))
print("\n")

# Calculate the absolute errors
errors = abs(y_predict - y_test)
# Print out the mean absolute error (mae)
print('Mean Absolute Error:', round(np.mean(errors), 2), 'kN.')

# Calculate mean absolute percentage error (MAPE)
mape = 100 * (errors / y_test)
# Calculate and display accuracy
accuracy = 100 - np.mean(mape)
print('Accuracy:', round(accuracy, 2), '%.')

# Commented out IPython magic to ensure Python compatibility.
# %matplotlib inline
import seaborn as sns
plt.figure(figsize=(5, 7))

ax = sns.distplot(y, hist=False, color='r', label='Actual Value')
sns.distplot(y_predict, hist=False, color='b', label='Fitted Values' ,
ax=ax)
```

```
plt.title('Actual vs Fitted Values')
ax.set(xlabel='Failure Load (kN)', ylabel='Density')
#ax.set(xlabel='Mid Span Deflection (in mm)', ylabel='Density')

plt.legend()
plt.show()
plt.close()

#============================ custom test
#Sl. 1
#model.predict([[3, 0, 0, 0, 0, 1, 31.21, 650.0, 13250, 250.0, 39.72,
454, 0]])
#sl.11,12 - phd work
#model.predict([[3, 0, 0, 0, 0, 1, 17.98, 1003.4, 13236, 420.8, 57.06,
415, 0]])
#model.predict([[3, 0, 0, 0, 0, 1, 17.98, 1003.4, 13236, 420.8, 57.06,
415, 0]])

model.predict([[1, 0, 0, 0, 1, 0, 29.8, 900, 13000, 0, 500, 420, 420]])

#model.predict([[3, 0, 0, 0, 1, 0, 31.21, 2650.0, 13250, 210.0, 100, 500,
165]])
#model.predict([[3, 0, 0, 0, 0, 1, 29.8, 900, 13000, 325, 56.55, 420,
0]])
#model.predict([[2, 0, 0, 0, 0, 0, 29.8, 900, 13000, 0, 0, 420, 0]])
#model.predict([[1, 0, 0, 0, 1, 0, 29.8, 900, 13000, 0, 600, 420, 420]])
#model.predict([[1, 0, 0, 0, 1, 0, 29.8, 900, 13000, 0, 1200, 420, 311]])

test =
pd.read_csv('/content/drive/MyDrive/Workshop/Aravind_CEE/test.csv')
print(test.shape)
test.head()
t_vect =
test.loc[:,['Shear_type','str_50','GFRP_full','U_50','U_Full','Steel_Shea
r','Com_con_MPa','Eff_span_mm','Xsect_area_sqmm','TS_stir_Mpa','Xsect_sti
r','TS_MS_Mpa','TS_gfrp_Mpa']]

model.predict(t_vect)
```

Abbreviations

L	effective span
P	monotonic loading
t_F	thickness of FRP strip
w_f	width of FRP strip
f_y	characteristic strength of steel
f_{fe}	ultimate tensile stress in the FRP shear reinforcement
S_f	spacing between the FRP strips
n	number of layers
A_{sv}	total cross sectional area of stirrups legs effective in shear
d	effective depth of the beam
S_v	Stirrups spacing along the length of the member
B	breadth of beam
D	overall depth of beam
RC	Reinforced Concrete
GFRP	Glass Fibre Reinforced Polymer
CFRP	Glass Fibre Reinforced Polymer
BFRP	Basalt Fibre Reinforced Polymer
SSWM	Stainless Steel Wire Mesh
CB	Control Beam
FA	Fine Aggregate
CA	Coarse Aggregate
ACI	American Concrete Institute
GSM	Gram per Square Meter
CSM	Chopped Strand Mat
WR	Woven Roving
LVDT	Linear Variable Differential Transformer
UTM	Universal Testing Machine
OPC	Ordinary Portland Cement
TRM	Textile Reinforced Mortar
EBRIG	Externally Bonded Reinforcement In Groove
BS EN	British Standard European Norm

ASTM	American Society for Testing and Materials
IS	Indian Standards
BPA	Bisphenol ‘A’
EB	Externally bonded Beam
SS	Strips on Sides
FS	Full on Sides
SU	U wrap Strip
FU	Full U wrap
Mpa	Mega Pascal
mm	millimeter
kN	kilo Newton
kg/m^3	kilo gram per meter cube
T	Tonne
RF	Random Forest
ML	Machine Learning
R^2	Coefficient of determination
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative

Theoretical Analysis Calculation

Calculation of shear resistance of a rectangular singly reinforced RC beam section externally bonded with GFRP CSM and GFRP laminates with the following data.

Overall beam cross section = 100 × 150 mm

Tension reinforcement = 2 – 8 mm diameter bars

Grade of steel = Fe 500

Grade of concrete = C30

Clear cover = 15 mm

Solution

a) Shear Strength of RC beam externally bonded with GFRP composites

Shear strength of RC beam externally bonded with GFRP composites,

$$V_u = \emptyset\, V_n = \emptyset\left(V_c + V_s + \psi_n V_f\right)$$

$w_f -$ width of GFRP CSM strip $= 50\ mm$

$t_f -$ thickness of GFRP CSM strip $= 0.5$ mm

$f_{fe} -$ tensile stress in the GFRP CSM shear reinforcement $= 166\ MPa$

$S_f -$ spacing between the FRP strips $= 100\ mm$

$n -$ number of layers $= 1$

$d_f -$ depth ofthe beam (GFRP composite attached full depth of beam) $= 150\ mm$

$\psi_n - additional\ reduction\ factors\ for\ FRP\ shear\ reinforcement =$ 0.85

$\emptyset -$ strength reduction factor

$= 0.7$ for brittle sections, as opposed to 0.90 for ductile sections.

b) Shear Strength of RC Beam

Based on IS 456, maximum shear stress for concrete is 3.5 MPa for 28 days cube compressive strength with 30 MPa. Width and effective depth of beam considered for the theoretical analysis are 100 mm and 131 mm respectively.

Effective depth, d = Overall depth – (clear cover + ϕ/2)

$$\mathrm{d} = 150 - \left(15 + {}^{8}/_{2}\right) = 131 \text{ mm}$$

$$\text{Shear resistance of concrete, } \mathrm{V}_c = \frac{2\ \mathrm{v}_{\max}\ bd}{3} = \frac{2 \times 3.5 \times 100 \times 131}{3} = 30566.67\ N$$

$$\mathrm{V}_c = 30.57\ kN$$

c) Shear contribution of steel shear reinforcement

Two legged 5 mm diameter steel shear reinforcements with yield stress 250 MPa at 175 mm spacing were provided.

The shear contribution of the steel shear reinforcement is then given by

$$\text{Shear resistance of steel stirrups, } \mathrm{V}_{\mathrm{us}} = \frac{0.87\ f_y\ A_{SV} d}{S_v}$$

$$\text{Area of steel stirrups, } \mathrm{A}_{\mathrm{st}} = 2\ \times 19.63 = 39.26 \text{ mm}^2$$

$$\mathrm{V}_{\mathrm{us}} = \frac{0.87 \times 250 \times 39.26 \times 131}{175} = 6392.09\ N$$

$$\mathrm{V}_{\mathrm{us}} = 6.39\ kN$$

d) Shear contribution of GFRP shear reinforcement

GFRP CSM composites with 50 mm strips at 100 mm spacing is attached on two sides of RC beam and tensile stress in GFRP is 166 MPa.

The shear contribution of the GFRP shear reinforcement is then given by

$$\mathrm{V}_f = \frac{\mathrm{A}_{fv}\ f_{fe}\ \mathrm{d}_f}{\mathrm{S}_f}$$

$$\text{Area of CSM strips bonded with both sides of RC beam, } \mathrm{A}_{\mathrm{fv}} = 2\ \mathrm{n}\ \mathrm{t}_{\mathrm{f}}\ \mathrm{w}_{\mathrm{f}}$$

$$\mathrm{A}_{\mathrm{fv}} = (2 \times 1 \times 0.5 \times 50) = 50\ mm^2$$

$$V_f = \frac{50 \times 166 \times 150}{100} = 12450\ N$$

$$V_f = 12.45\ kN$$

$$V_u = \emptyset\, V_n = \emptyset\,(V_c + V_s + \psi_n V_f)$$

Sample Designation	**Strength reduction factor (Ø)**	**Mechanical Property**
CB1	0.85	Ductile (RC beam with longitudinal steel reinforcement and steel stirrups)
CB2	0.80	Partially ductile (RC beam with longitudinal steel reinforcement without stirrups)
Beam bonded with CSM GFRP composites	0.75	Partially brittle (RC beam with longitudinal steel reinforcement with GFRP stirrups)

Shear resistance of CB1, $V_u = \emptyset\, V_n = \emptyset\,(V_c + V_s)$

$$V_u = \emptyset\, V_n = 0.85 \times (30.57 + 6.39) = 31.42\ kN$$

Shear resistance of CB2, $V_u = \emptyset\, V_n = \emptyset\,(V_c)$

$$V_u = \emptyset\, V_n = 0.80 \times 30.57 = 24.46\ kN$$

Shear resistance of RC beam with GFRP CSM strips on sides only ($EB - SS - CSM$),

$$V_u = \emptyset\, V_n = \emptyset\,(V_c + \psi_n V_f)$$

$$V_u = \emptyset\, V_n = 0.75 \times (30.57 + [0.85 \times 12.45]) = 30.86\ kN$$