Appendix_B_NN_and_DT_modeling

August 10, 2024

1 Neural Network and Decision Tree Modeling

This notebook contains the code and details for neural network and decision tree modeling.

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt


from pathlib import Path
from scikeras.wrappers import KerasClassifier
from sklearn.compose import ColumnTransformer
from sklearn.metrics import accuracy_score, confusion_matrix,␣
 ↪classification_report
from sklearn.model_selection import train_test_split, StratifiedShuffleSplit
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.tree import DecisionTreeClassifier, plot_tree
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.layers import Dense, Dropout, Input
from tensorflow.keras.models import Sequential
from tensorflow.keras.utils import to_categorical
```

[ ]:
```
# Import training dataset
dataset = Path('../dataset')

df = pd.read_csv(dataset/'accidents_clean_train.csv')

df.head()
```

[ ]:
```
  Area_accident_occured Types_of_Junction       Light_conditions  \
0     Residential areas       No junction               Daylight
1          Office areas       No junction               Daylight
2    Recreational areas       No junction               Daylight
3          Office areas           Y Shape  Darkness - lights lit
4      Industrial areas           Y Shape  Darkness - lights lit

   Number_of_vehicles_involved  Number_of_casualties  \
0                            2                     2
```

```
1                            2                     2
2                            2                     2
3                            2                     2
4                            2                     2

             Cause_of_accident Day_of_week Sex_of_driver Age_band_of_driver  \
0              Moving Backward      Monday          Male              18-30
1                   Overtaking      Monday          Male              31-50
2    Changing lane to the left      Monday          Male              18-30
3   Changing lane to the right      Sunday          Male              18-30
4                   Overtaking      Sunday          Male              18-30

  Accident_severity
0    Slight Injury
1    Slight Injury
2   Serious Injury
3    Slight Injury
4    Slight Injury
```

[ ]:

```
%run ../custom/jc-functions.ipynb
```

2 Decision Tree

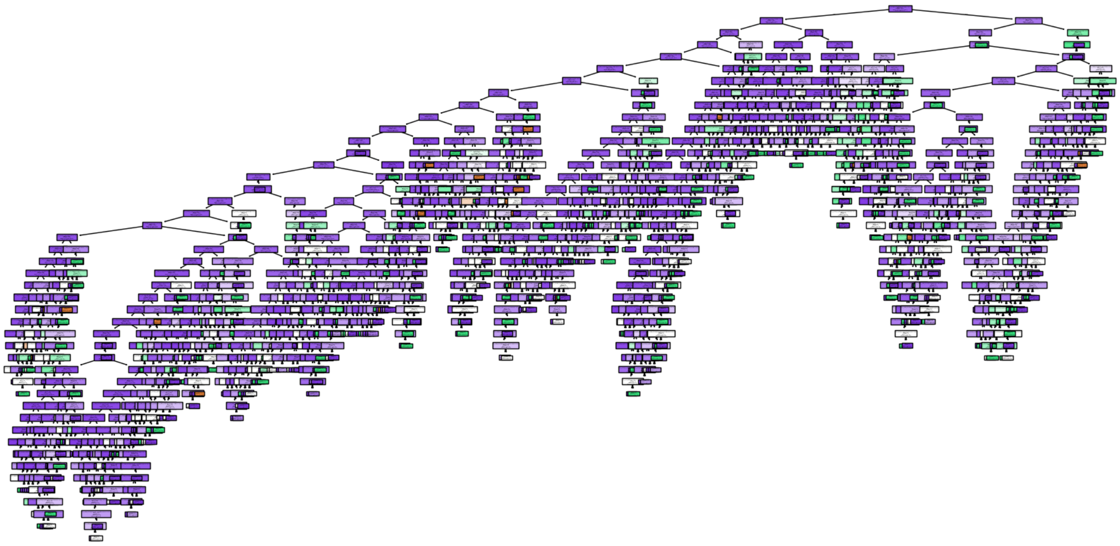
Slight injury (2) vs Serious injury (1) vs Fatal injury (0)

[ ]:

```
label_encoder = LabelEncoder()
df['Accident_severity'] = label_encoder.fit_transform(df['Accident_severity'])

X = df.drop(columns=['Accident_severity'])
y = df['Accident_severity']


categorical_variables = X.columns.tolist()


preprocessor = ColumnTransformer(
    transformers=[('cat', OneHotEncoder(), categorical_variables)]
)

X_encoded = preprocessor.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.
 ↪3, random_state=42, stratify=y)


cart_model = DecisionTreeClassifier(random_state=42)
cart_model.fit(X_train, y_train)
```

```
y_pred = cart_model.predict(X_test)


def cart_report(test, pred):
    print("Accuracy: ", accuracy_score(test, pred))
    print("Confusion Matrix:\n", confusion_matrix(test, pred))
    print("Classification Report:\n", classification_report(test, pred,␣
 ↪zero_division=0))


cart_report(y_test, y_pred)


plt.figure(figsize=(20,10))
plot_tree(cart_model, filled=True, feature_names=preprocessor.
 ↪get_feature_names_out(), class_names=label_encoder.classes_.astype(str))
plt.show()
```

```
Accuracy:  0.7840032480714576
Confusion Matrix:
 [[   7    3   14]
 [   1   84  229]
 [  14  271 1840]]
Classification Report:
               precision    recall  f1-score   support

           0       0.32      0.29      0.30        24
           1       0.23      0.27      0.25       314
           2       0.88      0.87      0.87      2125

    accuracy                           0.78      2463
   macro avg       0.48      0.48      0.48      2463
weighted avg       0.80      0.78      0.79      2463
```



```
[ ]: decoded_labels = label_encoder.inverse_transform(df['Accident_severity'])
     print(decoded_labels)
```

```
['Slight Injury' 'Slight Injury' 'Serious Injury' … 'Slight Injury'
 'Slight Injury' 'Slight Injury']
```

```
[ ]: df['Accident_severity']
     df['Accident_severity'].unique()
```

```
[ ]: array([2, 1, 0])
```

Decoding target variable:

- Slight Injury = 2
- Serious Injury = 1
- Fatal Injury = 0

```
[ ]: df.head()
```

```
[ ]:   Area_accident_occured Types_of_Junction       Light_conditions  \
     0     Residential areas       No junction               Daylight
     1          Office areas       No junction               Daylight
     2    Recreational areas       No junction               Daylight
     3          Office areas           Y Shape  Darkness - lights lit
     4      Industrial areas           Y Shape  Darkness - lights lit

        Number_of_vehicles_involved  Number_of_casualties  \
     0                            2                     2
     1                            2                     2
     2                            2                     2
```

```
3                             2                     2
4                             2                     2

            Cause_of_accident Day_of_week Sex_of_driver Age_band_of_driver  \
0             Moving Backward      Monday          Male              18-30
1                  Overtaking      Monday          Male              31-50
2    Changing lane to the left      Monday          Male              18-30
3   Changing lane to the right      Sunday          Male              18-30
4                  Overtaking      Sunday          Male              18-30

   Accident_severity
0                  2
1                  2
2                  1
3                  2
4                  2
```

[ ]:
```
dtree_accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {dtree_accuracy}")
dtree_cm = confusion_matrix(y_test, y_pred)
print(multiclass_cm_metrics(dtree_cm))
```

```
Accuracy: 0.7840032480714576
Confusion Matrix:
[[   7    3   14]
 [   1   84  229]
 [  14  271 1840]]
                      Class 0  Class 1  Class 2
Accuracy              0.98701  0.79537  0.78563
Error rate            0.01299  0.20463  0.21437
Sensitivity (Recall)  0.29167  0.26752  0.86588
Specificity           0.99385  0.87250  0.28107
Precision             0.31818  0.23464  0.88334
F1                    0.30435  0.25000  0.87452
F2                    0.29661  0.26022  0.86932
F0.5                  0.31250  0.24055  0.87979
```

3 Neural Network

[ ]:
```
X.head()
```

[ ]:
```
  Area_accident_occured Types_of_Junction       Light_conditions  \
0     Residential areas       No junction               Daylight
1          Office areas       No junction               Daylight
2    Recreational areas       No junction               Daylight
3          Office areas           Y Shape  Darkness - lights lit
4      Industrial areas           Y Shape  Darkness - lights lit
```

```
   Number_of_vehicles_involved  Number_of_casualties  \
0                            2                     2
1                            2                     2
2                            2                     2
3                            2                     2
4                            2                     2

            Cause_of_accident Day_of_week Sex_of_driver Age_band_of_driver
0             Moving Backward      Monday          Male              18-30
1                  Overtaking      Monday          Male              31-50
2   Changing lane to the left      Monday          Male              18-30
3  Changing lane to the right      Sunday          Male              18-30
4                  Overtaking      Sunday          Male              18-30
```

[ ]:

```
# label_encoder = LabelEncoder()
# df['Accident_severity'] = label_encoder.fit_transform(df['Accident_severity'])


X = df.drop(columns=['Accident_severity'])
y = df['Accident_severity']


categorical_variables = X.columns.tolist()


preprocessor = ColumnTransformer(
    transformers=[('cat', OneHotEncoder(), categorical_variables)]
)


X_encoded = preprocessor.fit_transform(X)


def create_model(optimizer='adam', activation='relu', dropout_rate=0.5):
    model = Sequential()
    model.add(Input(shape=(X_encoded.shape[1],)))
    model.add(Dense(32, activation=activation))
    model.add(Dropout(dropout_rate))
    model.add(Dense(16, activation=activation))
    model.add(Dropout(dropout_rate))
    model.add(Dense(y_train_encoded.shape[1], activation='softmax'))
    model.compile(optimizer=optimizer, loss='categorical_crossentropy',␣
 ↪metrics=['accuracy'])
    return model
```

```
model = KerasClassifier(model=create_model, verbose=0)


def nn_report(test, pred):
    print("Accuracy: ", accuracy_score(test, pred))
    print("Confusion Matrix:\n", confusion_matrix(test, pred))
    print("Classification Report:\n", classification_report(test, pred))

def cross_scores(score):
    print("Cross-validation scores: ", score)
    print("Average score: ", score.mean())

early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

sss = StratifiedShuffleSplit(n_splits=5, test_size=0.3, random_state=42)
scores = []

for train_index, test_index in sss.split(X_encoded, y):
    X_train, X_test = X_encoded[train_index], X_encoded[test_index]
    y_train, y_test = y[train_index], y[test_index]
    y_train_encoded = to_categorical(y_train)
    y_test_encoded = to_categorical(y_test)

    nn_model = create_model()
    nn_model.fit(X_train, y_train_encoded, epochs=50, batch_size=32, verbose=1, validation_split=0.2, callbacks=[early_stopping])

    y_pred_prob = nn_model.predict(X_test)
    y_pred = np.argmax(y_pred_prob, axis=1)

    scores.append(accuracy_score(y_test, y_pred))
    nn_report(y_test, y_pred)

cross_scores(np.array(scores))
```

```
Epoch 1/50
144/144             1s 3ms/step -
accuracy: 0.4694 - loss: 1.1009 - val_accuracy: 0.8696 - val_loss: 0.4470
Epoch 2/50
144/144             0s 1ms/step -
accuracy: 0.8458 - loss: 0.5879 - val_accuracy: 0.8696 - val_loss: 0.4404
Epoch 3/50
144/144             0s 1ms/step -
accuracy: 0.8496 - loss: 0.5716 - val_accuracy: 0.8696 - val_loss: 0.4188
Epoch 4/50
144/144             0s 2ms/step -
```

```
accuracy: 0.8568 - loss: 0.5288 - val_accuracy: 0.8696 - val_loss: 0.4156
Epoch 5/50
144/144             0s 2ms/step -
accuracy: 0.8677 - loss: 0.4752 - val_accuracy: 0.8696 - val_loss: 0.4185
Epoch 6/50
144/144             0s 1ms/step -
accuracy: 0.8591 - loss: 0.4736 - val_accuracy: 0.8696 - val_loss: 0.4141
Epoch 7/50
144/144             0s 2ms/step -
accuracy: 0.8661 - loss: 0.4646 - val_accuracy: 0.8696 - val_loss: 0.4120
Epoch 8/50
144/144             0s 1ms/step -
accuracy: 0.8619 - loss: 0.4548 - val_accuracy: 0.8696 - val_loss: 0.4118
Epoch 9/50
144/144             0s 1ms/step -
accuracy: 0.8637 - loss: 0.4564 - val_accuracy: 0.8696 - val_loss: 0.4155
Epoch 10/50
144/144             0s 1ms/step -
accuracy: 0.8628 - loss: 0.4589 - val_accuracy: 0.8696 - val_loss: 0.4146
Epoch 11/50
144/144             0s 1ms/step -
accuracy: 0.8624 - loss: 0.4484 - val_accuracy: 0.8696 - val_loss: 0.4120
Epoch 12/50
144/144             0s 1ms/step -
accuracy: 0.8544 - loss: 0.4603 - val_accuracy: 0.8696 - val_loss: 0.4110
Epoch 13/50
144/144             0s 1ms/step -
accuracy: 0.8633 - loss: 0.4379 - val_accuracy: 0.8696 - val_loss: 0.4116
Epoch 14/50
144/144             0s 1ms/step -
accuracy: 0.8712 - loss: 0.4361 - val_accuracy: 0.8696 - val_loss: 0.4115
Epoch 15/50
144/144             0s 1ms/step -
accuracy: 0.8583 - loss: 0.4598 - val_accuracy: 0.8696 - val_loss: 0.4103
Epoch 16/50
144/144             0s 1ms/step -
accuracy: 0.8655 - loss: 0.4393 - val_accuracy: 0.8696 - val_loss: 0.4107
Epoch 17/50
144/144             0s 1ms/step -
accuracy: 0.8616 - loss: 0.4383 - val_accuracy: 0.8696 - val_loss: 0.4102
Epoch 18/50
144/144             0s 2ms/step -
accuracy: 0.8561 - loss: 0.4485 - val_accuracy: 0.8696 - val_loss: 0.4095
Epoch 19/50
144/144             0s 2ms/step -
accuracy: 0.8630 - loss: 0.4375 - val_accuracy: 0.8696 - val_loss: 0.4096
Epoch 20/50
144/144             0s 1ms/step -
```

```
accuracy: 0.8667 - loss: 0.4231 - val_accuracy: 0.8696 - val_loss: 0.4093
Epoch 21/50
144/144            0s 1ms/step -
accuracy: 0.8673 - loss: 0.4159 - val_accuracy: 0.8696 - val_loss: 0.4088
Epoch 22/50
144/144            0s 1ms/step -
accuracy: 0.8662 - loss: 0.4343 - val_accuracy: 0.8696 - val_loss: 0.4100
Epoch 23/50
144/144            0s 1ms/step -
accuracy: 0.8603 - loss: 0.4300 - val_accuracy: 0.8696 - val_loss: 0.4084
Epoch 24/50
144/144            0s 2ms/step -
accuracy: 0.8612 - loss: 0.4234 - val_accuracy: 0.8696 - val_loss: 0.4085
Epoch 25/50
144/144            0s 2ms/step -
accuracy: 0.8651 - loss: 0.4143 - val_accuracy: 0.8696 - val_loss: 0.4091
Epoch 26/50
144/144            0s 2ms/step -
accuracy: 0.8669 - loss: 0.4119 - val_accuracy: 0.8696 - val_loss: 0.4077
Epoch 27/50
144/144            0s 1ms/step -
accuracy: 0.8520 - loss: 0.4401 - val_accuracy: 0.8696 - val_loss: 0.4081
Epoch 28/50
144/144            0s 1ms/step -
accuracy: 0.8540 - loss: 0.4362 - val_accuracy: 0.8696 - val_loss: 0.4102
Epoch 29/50
144/144            0s 2ms/step -
accuracy: 0.8561 - loss: 0.4319 - val_accuracy: 0.8696 - val_loss: 0.4083
Epoch 30/50
144/144            0s 1ms/step -
accuracy: 0.8564 - loss: 0.4286 - val_accuracy: 0.8696 - val_loss: 0.4079
Epoch 31/50
144/144            0s 1ms/step -
accuracy: 0.8535 - loss: 0.4420 - val_accuracy: 0.8696 - val_loss: 0.4084
77/77            0s 2ms/step
Accuracy:  0.8627689809175801
Confusion Matrix:
 [[   0    0   24]
 [   0    0  314]
 [   0    0 2125]]
Classification Report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        24
           1       0.00      0.00      0.00       314
           2       0.86      1.00      0.93      2125

    accuracy                           0.86      2463
```

```
   macro avg       0.29      0.33      0.31      2463
weighted avg       0.74      0.86      0.80      2463

Epoch 1/50

c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))

144/144            1s 3ms/step -
accuracy: 0.3578 - loss: 1.1177 - val_accuracy: 0.8643 - val_loss: 0.4930
Epoch 2/50
144/144            0s 2ms/step -
accuracy: 0.8501 - loss: 0.6315 - val_accuracy: 0.8643 - val_loss: 0.4354
Epoch 3/50
144/144            0s 2ms/step -
accuracy: 0.8533 - loss: 0.5601 - val_accuracy: 0.8643 - val_loss: 0.4292
Epoch 4/50
144/144            0s 2ms/step -
accuracy: 0.8532 - loss: 0.5308 - val_accuracy: 0.8643 - val_loss: 0.4277
Epoch 5/50
144/144            0s 2ms/step -
accuracy: 0.8558 - loss: 0.5141 - val_accuracy: 0.8643 - val_loss: 0.4233
77/77            0s 2ms/step
Accuracy:  0.8627689809175801
Confusion Matrix:
 [[   0    0   24]
 [   0    0  314]
 [   0    0 2125]]
Classification Report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        24
           1       0.00      0.00      0.00       314
           2       0.86      1.00      0.93      2125

    accuracy                           0.86      2463
   macro avg       0.29      0.33      0.31      2463
weighted avg       0.74      0.86      0.80      2463
```

```
Epoch 1/50
```

```
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
144/144            1s 3ms/step -
accuracy: 0.7166 - loss: 0.8066 - val_accuracy: 0.8696 - val_loss: 0.4393
Epoch 2/50
144/144            0s 2ms/step -
accuracy: 0.8413 - loss: 0.5675 - val_accuracy: 0.8696 - val_loss: 0.4199
Epoch 3/50
144/144            0s 2ms/step -
accuracy: 0.8528 - loss: 0.5325 - val_accuracy: 0.8696 - val_loss: 0.4163
Epoch 4/50
144/144            0s 2ms/step -
accuracy: 0.8572 - loss: 0.5082 - val_accuracy: 0.8696 - val_loss: 0.4161
Epoch 5/50
144/144            0s 2ms/step -
accuracy: 0.8574 - loss: 0.4882 - val_accuracy: 0.8696 - val_loss: 0.4167
77/77            0s 3ms/step
Accuracy:  0.8627689809175801
Confusion Matrix:
 [[   0    0   24]
 [   0    0  314]
 [   0    0 2125]]
Classification Report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        24
           1       0.00      0.00      0.00       314
           2       0.86      1.00      0.93      2125

    accuracy                           0.86      2463
```

```
   macro avg       0.29      0.33      0.31      2463
weighted avg       0.74      0.86      0.80      2463

Epoch 1/50

c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))

144/144            1s 3ms/step -
accuracy: 0.6247 - loss: 0.8972 - val_accuracy: 0.8661 - val_loss: 0.4407
Epoch 2/50
144/144            0s 2ms/step -
accuracy: 0.8512 - loss: 0.5605 - val_accuracy: 0.8661 - val_loss: 0.4251
Epoch 3/50
144/144            0s 1ms/step -
accuracy: 0.8540 - loss: 0.5234 - val_accuracy: 0.8661 - val_loss: 0.4246
Epoch 4/50
144/144            0s 2ms/step -
accuracy: 0.8610 - loss: 0.4947 - val_accuracy: 0.8661 - val_loss: 0.4269
Epoch 5/50
144/144            0s 2ms/step -
accuracy: 0.8601 - loss: 0.4807 - val_accuracy: 0.8661 - val_loss: 0.4258
77/77            0s 2ms/step
Accuracy:  0.8627689809175801
Confusion Matrix:
 [[   0    0   24]
 [   0    0  314]
 [   0    0 2125]]
Classification Report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        24
           1       0.00      0.00      0.00       314
           2       0.86      1.00      0.93      2125

    accuracy                           0.86      2463
   macro avg       0.29      0.33      0.31      2463
weighted avg       0.74      0.86      0.80      2463


c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
Epoch 1/50
144/144             1s 3ms/step -
accuracy: 0.7452 - loss: 0.7222 - val_accuracy: 0.8365 - val_loss: 0.5084
Epoch 2/50
144/144            0s 2ms/step -
accuracy: 0.8568 - loss: 0.5145 - val_accuracy: 0.8365 - val_loss: 0.4950
Epoch 3/50
144/144            0s 2ms/step -
accuracy: 0.8619 - loss: 0.4914 - val_accuracy: 0.8365 - val_loss: 0.4936
Epoch 4/50
144/144            0s 2ms/step -
accuracy: 0.8730 - loss: 0.4493 - val_accuracy: 0.8365 - val_loss: 0.4886
Epoch 5/50
144/144            0s 2ms/step -
accuracy: 0.8779 - loss: 0.4331 - val_accuracy: 0.8365 - val_loss: 0.4890
77/77             0s 2ms/step
Accuracy:  0.8627689809175801
Confusion Matrix:
 [[   0    0   24]
 [   0    0  314]
 [   0    0 2125]]
Classification Report:
               precision    recall  f1-score   support

           0       0.00      0.00      0.00        24
           1       0.00      0.00      0.00       314
           2       0.86      1.00      0.93      2125

    accuracy                           0.86      2463
```

```
   macro avg       0.29      0.33      0.31      2463
weighted avg       0.74      0.86      0.80      2463

Cross-validation scores:  [0.86276898 0.86276898 0.86276898 0.86276898
0.86276898]
Average score:  0.8627689809175803
```

```
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
c:\Users\xxkjx\miniconda3\envs\mambaML\lib\site-
packages\sklearn\metrics\_classification.py:1531: UndefinedMetricWarning:
Precision is ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

3.0.1 Performance Metrics of last neural network iteration

Decoding target variable:

- Slight Injury = 2
- Serious Injury = 1
- Fatal Injury = 0

```
[ ]: nn_accuracy = accuracy_score(y_test, y_pred)
     print(f"Accuracy: {nn_accuracy}")
     neural_cm = confusion_matrix(y_test, y_pred)
     nn_metrics = multiclass_cm_metrics(neural_cm)
     print(nn_metrics)
```

```
Accuracy: 0.8627689809175801
Confusion Matrix:
[[   0    0   24]
 [   0    0  314]
 [   0    0 2125]]
                      Class 0  Class 1  Class 2
Accuracy              0.99026  0.87251  0.86277
Error rate            0.00974  0.12749  0.13723
Sensitivity (Recall)  0.00000  0.00000  1.00000
Specificity           1.00000  1.00000  0.00000
Precision             0.00000  0.00000  0.86277
F1                    0.00000  0.00000  0.92633
```

```
F2                      0.00000  0.00000  0.96917
F0.5                    0.00000  0.00000  0.88712
```

```
[ ]:
```