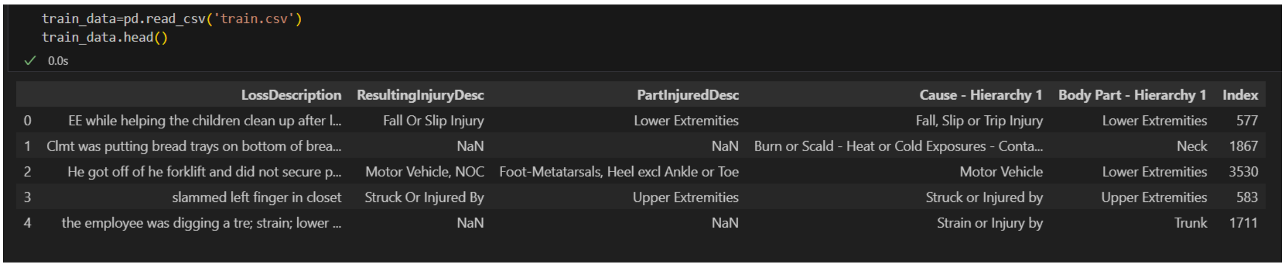
PROJECT REPORT

INTRODUCTION

The following report compiles all the steps involved in pre-processing, modelling, evaluation, prediction and analysis.

DATA OVERVIEW & PREPROCESSING



```
train_data=pd.read_csv('train.csv')
train_data.head()
```

✓ 0.0s

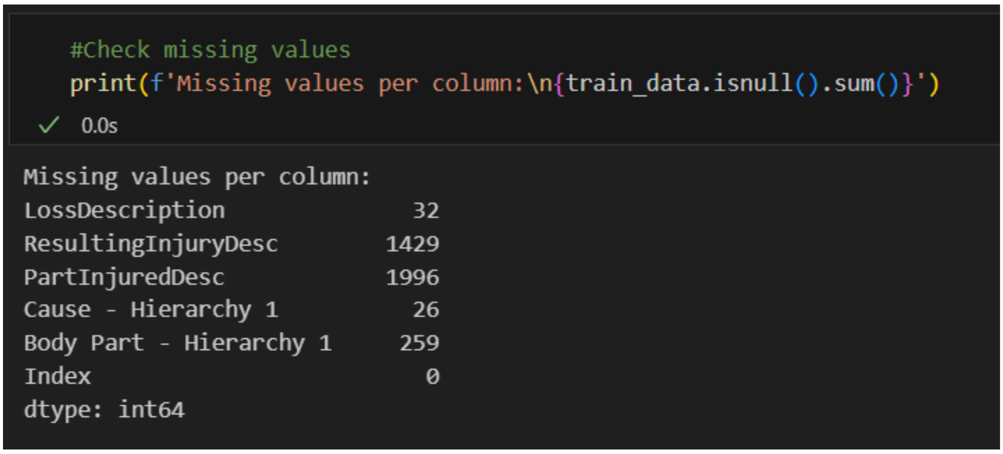
	LossDescription	ResultingInjuryDesc	PartInjuredDesc	Cause - Hierarchy 1	Body Part - Hierarchy 1	Index
0	EE while helping the children clean up after l...	Fall Or Slip Injury	Lower Extremities	Fall, Slip or Trip Injury	Lower Extremities	577
1	Clmt was putting bread trays on bottom of brea...	NaN	NaN	Burn or Scald - Heat or Cold Exposures - Conta...	Neck	1867
2	He got off of he forklift and did not secure p...	Motor Vehicle, NOC	Foot-Metatarsals, Heel excl Ankle or Toe	Motor Vehicle	Lower Extremities	3530
3	slammed left finger in closet	Struck Or Injured By	Upper Extremities	Struck or Injured by	Upper Extremities	583
4	the employee was digging a tre; strain; lower ...	NaN	NaN	Strain or Injury by	Trunk	1711

- Number of rows: 3918
- Number of columns: 6

MISSING VALUES:

After examining the dataset, the following columns contain missing values:

- LossDescription: 32 missing values
- ResultingInjuryDesc: 1429 missing values
- PartInjuredDesc: 1996 missing values
- Cause - Hierarchy 1: 26 missing values
- Body Part - Hierarchy 1: 259 missing values



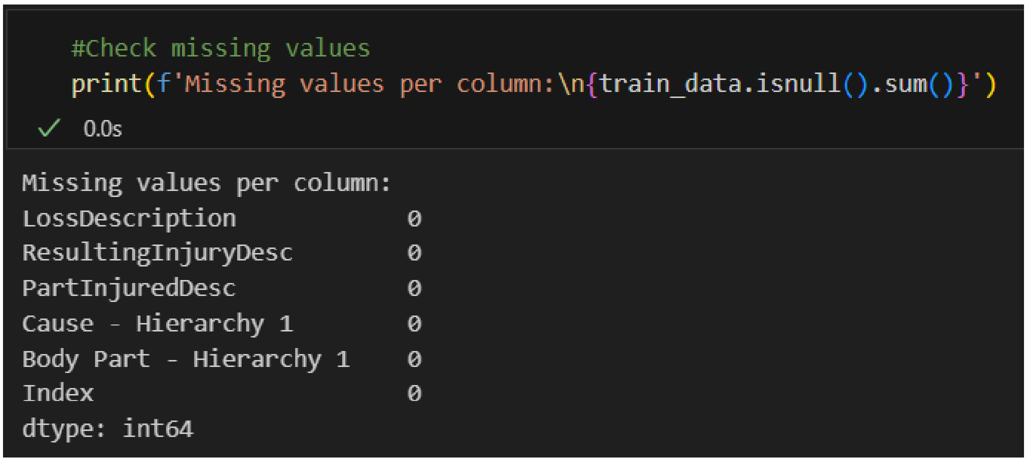
```
#Check missing values
print(f'Missing values per column:\n{train_data.isnull().sum()}')
```

✓ 0.0s

```
Missing values per column:
LossDescription              32
ResultingInjuryDesc        1429
PartInjuredDesc            1996
Cause - Hierarchy 1          26
Body Part - Hierarchy 1     259
Index                         0
dtype: int64
```

To address missing values, the following strategy was applied:

- **Fill missing values:** Missing values in the text columns ('LossDescription', 'ResultingInjuryDesc', 'PartInjuredDesc', 'Cause - Hierarchy 1', 'Body Part - Hierarchy 1') were filled with the string 'None'.



```
#Check missing values
print(f'Missing values per column:\n{train_data.isnull().sum()}')
```

```
Missing values per column:
LossDescription          0
ResultingInjuryDesc      0
PartInjuredDesc          0
Cause - Hierarchy 1      0
Body Part - Hierarchy 1  0
Index                    0
dtype: int64
```

DATA TYPES:

The data types of the columns in the dataset are as follows:

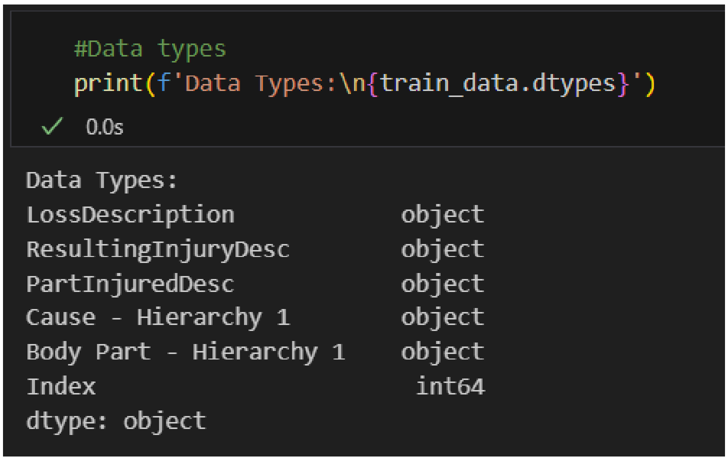
- LossDescription: Object (text)
- ResultingInjuryDesc: Object (text)
- PartInjuredDesc: Object (text)
- Cause - Hierarchy 1: Object (text)
- Body Part - Hierarchy 1: Object (text)
- Index: Integer

UNIQUE VALUES:

Unique values in the 'Cause - Hierarchy 1' and 'Body Part - Hierarchy 1' columns are as follows:

Unique Values in 'Cause - Hierarchy 1':

- Struck or Injured by: 727 occurrences
- Strain or Injury by: 715 occurrences
- Misc.: 700 occurrences
- Fall, Slip or Trip Injury: 669 occurrences
- ... (other unique values)



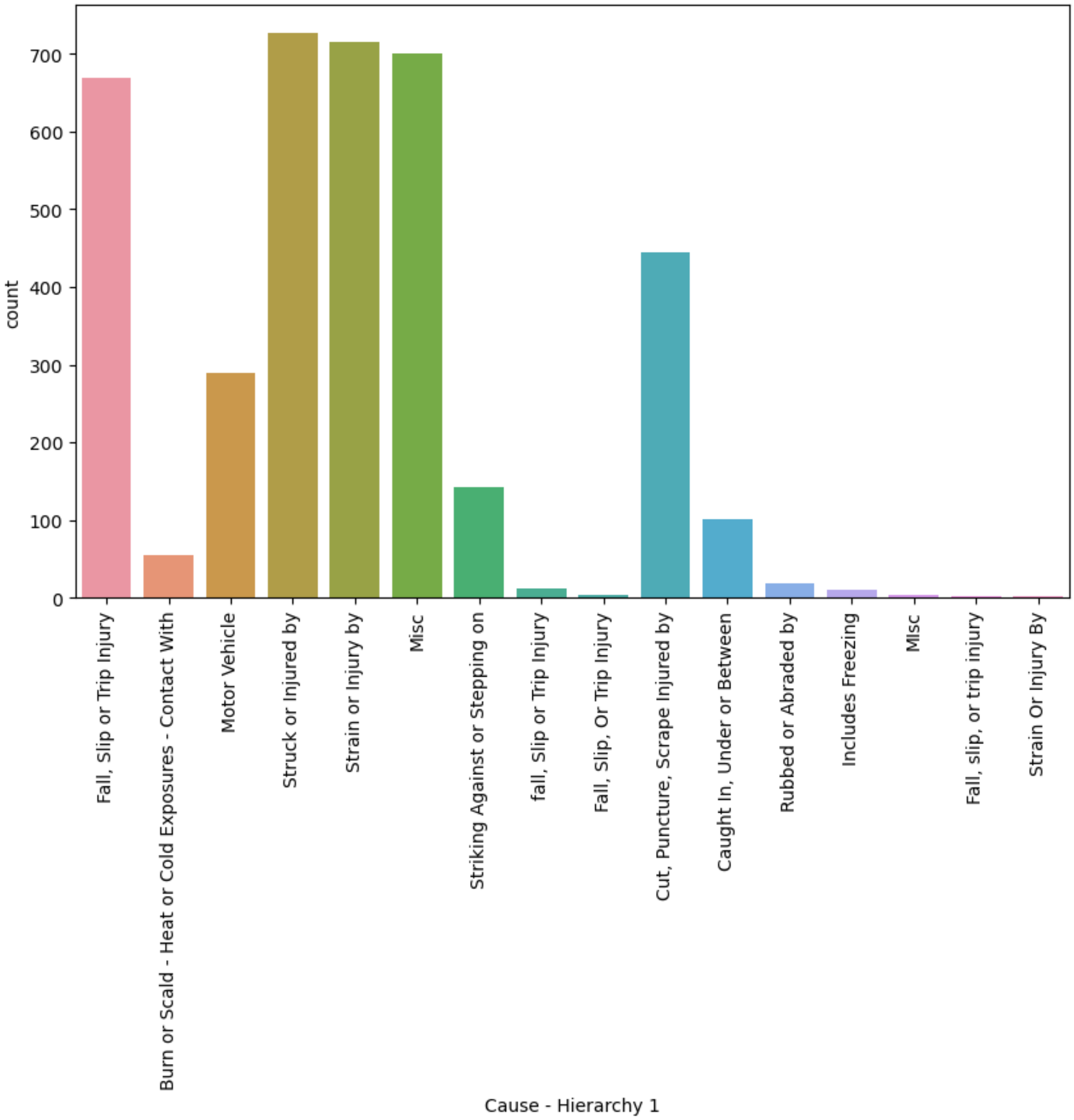
```
#Data types
print(f'Data Types:\n{train_data.dtypes}')
```

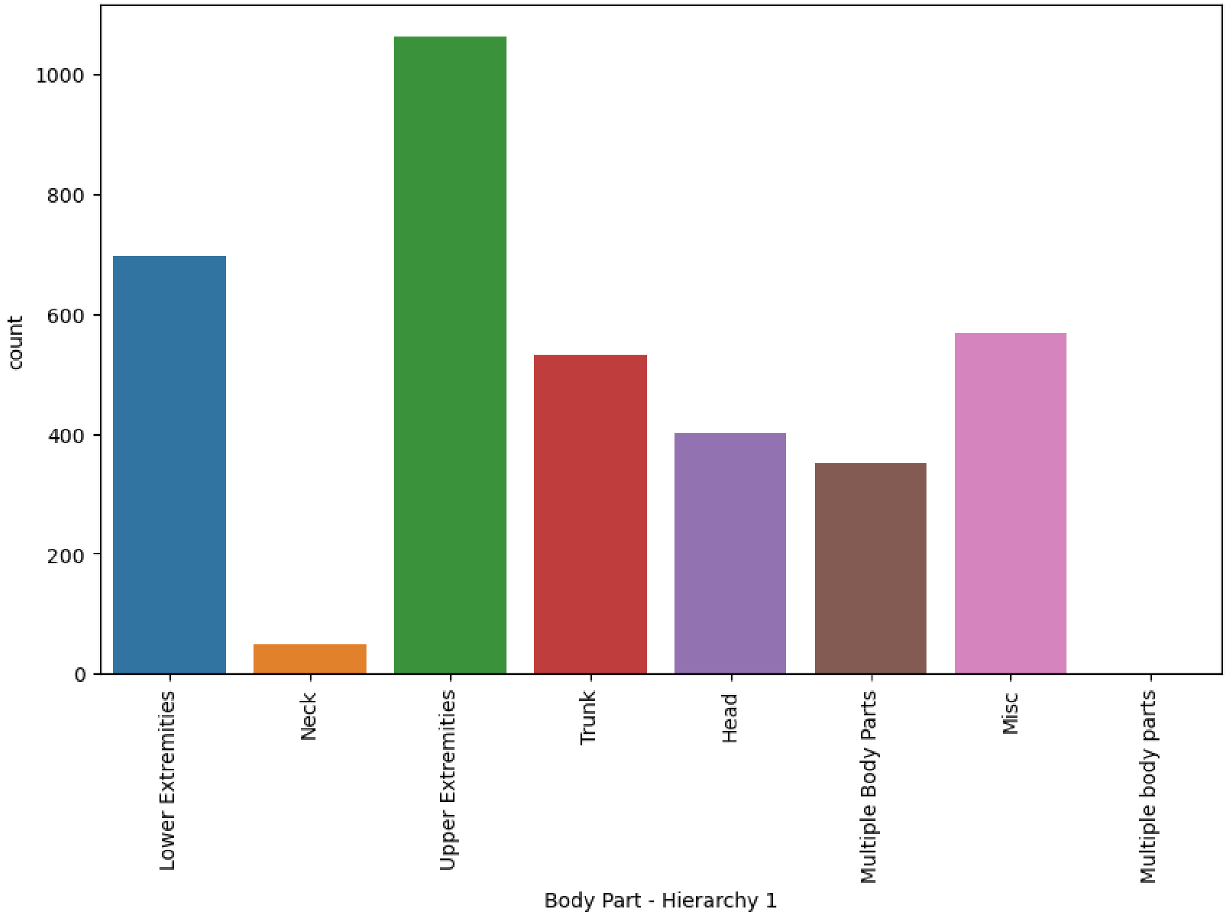
```
Data Types:
LossDescription            object
ResultingInjuryDesc        object
PartInjuredDesc            object
Cause - Hierarchy 1        object
Body Part - Hierarchy 1    object
Index                       int64
dtype: object
```

Unique Values in 'Body Part - Hierarchy 1':

- Upper Extremities: 1062 occurrences

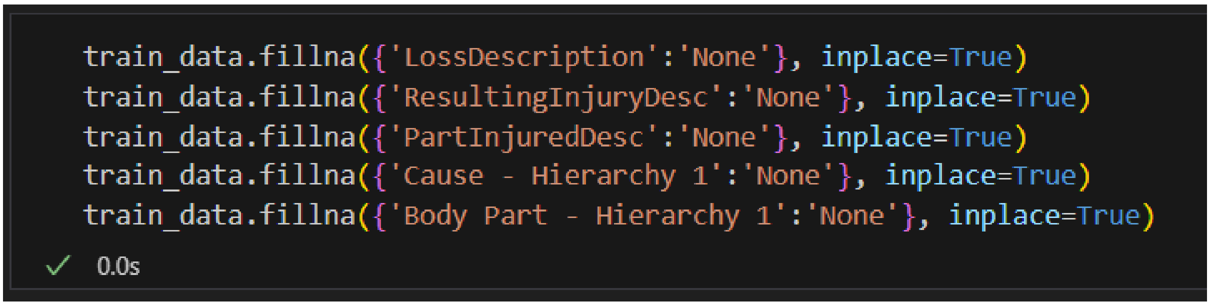
- Lower Extremities: 695 occurrences
- Misc.: 569 occurrences
- Trunk: 531 occurrences
- ... (other unique values)





PRE-PROCESSING STEPS:

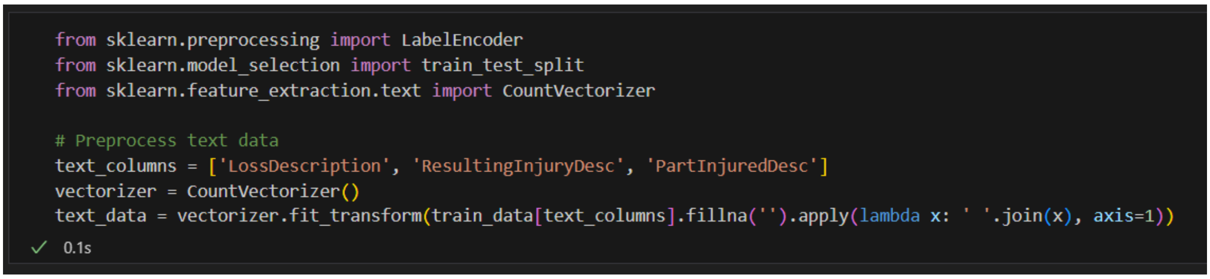
- Handling Missing Values:
- Missing values in text columns were filled with the string 'None' using the fillna() method.



```
train_data.fillna({'LossDescription':'None'}, inplace=True)
train_data.fillna({'ResultingInjuryDesc':'None'}, inplace=True)
train_data.fillna({'PartInjuredDesc':'None'}, inplace=True)
train_data.fillna({'Cause - Hierarchy 1':'None'}, inplace=True)
train_data.fillna({'Body Part - Hierarchy 1':'None'}, inplace=True)
```

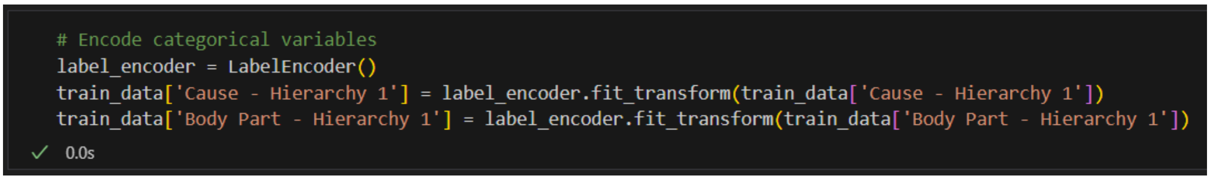
✓ 0.0s

- Text Data Pre-Processing:
  - Text data from columns 'LossDescription', 'ResultingInjuryDesc', and 'PartInjuredDesc' were pre-processed using CountVectorizer to convert them into a numerical format suitable for machine learning models.



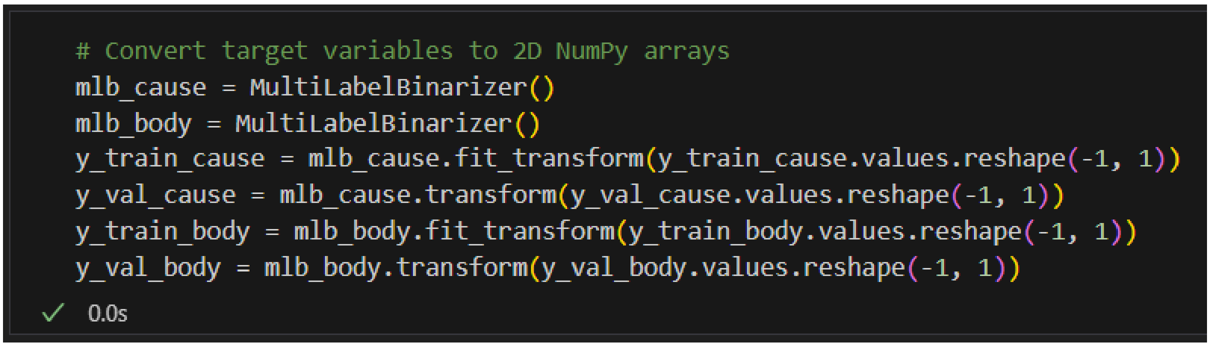
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer

# Preprocess text data
text_columns = ['LossDescription', 'ResultingInjuryDesc', 'PartInjuredDesc']
vectorizer = CountVectorizer()
text_data = vectorizer.fit_transform(train_data[text_columns].fillna('').apply(lambda x: ' '.join(x), axis=1))
✓ 0.1s
```

- Categorical Variable Encoding:
  - Categorical variables 'Cause - Hierarchy 1' and 'Body Part - Hierarchy 1' were encoded using LabelEncoder from scikit-learn.



```
# Encode categorical variables
label_encoder = LabelEncoder()
train_data['Cause - Hierarchy 1'] = label_encoder.fit_transform(train_data['Cause - Hierarchy 1'])
train_data['Body Part - Hierarchy 1'] = label_encoder.fit_transform(train_data['Body Part - Hierarchy 1'])
✓ 0.0s
```

- Train-Validation Split:
  - The dataset was split into training and validation sets using a test size of 20% and a random state = 42 for reproducibility.
- Target Variables Transformation:
  - The target variables 'Cause - Hierarchy 1' and 'Body Part - Hierarchy 1' were transformed into a binary format using MultiLabelBinarizer to prepare them for model training.



```
# Convert target variables to 2D NumPy arrays
mlb_cause = MultiLabelBinarizer()
mlb_body = MultiLabelBinarizer()
y_train_cause = mlb_cause.fit_transform(y_train_cause.values.reshape(-1, 1))
y_val_cause = mlb_cause.transform(y_val_cause.values.reshape(-1, 1))
y_train_body = mlb_body.fit_transform(y_train_body.values.reshape(-1, 1))
y_val_body = mlb_body.transform(y_val_body.values.reshape(-1, 1))
✓ 0.0s
```

MODELS AND EVALUATION:

The pre-processed data was then used to train five different models. The models with their respective evaluation on validation data has been shown as follows:

Binary Relevance with Logistic Regression:

- Cause - Hierarchy 1 Evaluation Metrics (by Binary Relevance):
  - Accuracy: 0.7602
  - Precision: 0.9099
  - Recall: 0.7739
  - F1-score: 0.8364
- Body Part - Hierarchy 1 Evaluation Metrics (by Binary Relevance):
  - Accuracy: 0.8010
  - Precision: 0.9281
  - Recall: 0.8227

  - F1-score: 0.8722

Multi-Output Classifier with k-Nearest Neighbors (k-NN):

- Cause - Hierarchy 1 Evaluation Metrics (by KNN):
  - Accuracy: 0.6594
  - Precision: 0.8617
  - Recall: 0.6603
  - F1-score: 0.7477
- Body Part - Hierarchy 1 Evaluation Metrics (by KNN):
  - Accuracy: 0.5408
  - Precision: 0.7896
  - Recall: 0.5408
  - F1-score: 0.6419

Classifier Chains with Logistic Regression:

- Cause - Hierarchy 1 Evaluation Metrics (by Classifier Chain):
  - Accuracy: 0.8163
  - Precision: 0.8677
  - Recall: 0.8212
  - F1-score: 0.8438
- Body Part - Hierarchy 1 Evaluation Metrics (by Classifier Chain):
  - Accuracy: 0.8418
  - Precision: 0.8711
  - Recall: 0.8444
  - F1-score: 0.8575

Multi-Output Classifier with Decision Tree:

- Cause - Hierarchy 1 Evaluation Metrics (by Decision Tree Classifier):
  - Accuracy: 0.7079
  - Precision: 0.8043
  - Recall: 0.8135
  - F1-score: 0.8089
- Body Part - Hierarchy 1 Evaluation Metrics (by Decision Tree Classifier):
  - Accuracy: 0.7653
  - Precision: 0.8597
  - Recall: 0.8520
  - F1-score: 0.8558

Multi-Output Neural Network:

- Cause - Hierarchy 1 Evaluation Metrics (by Neural Network):
  - Accuracy: 0.8418
  - Precision: 0.9135
  - Recall: 0.7688
  - F1-score: 0.8350
- Body Part - Hierarchy 1 Evaluation Metrics (by Neural Network):
  - Accuracy: 0.8852
  - Precision: 0.9360

- Recall: 0.8201
- F1-score: 0.8742

Considering the F1-score of each model, from the evaluation results, we can see that:

For “Cause – Hierarchy 1”, the Binary Relevance with Logistic Regression has the highest F1-score. Hence, for prediction of test data, this model will be chosen.

For “Body Part – Hierarchy 1”, the Neural Network has the highest F1-score. Hence, for prediction of test data, this model will be chosen.

PREDICTION AND ANALYSIS:

The predictions.csv file contains the predicted outputs, ‘Cause - Hierarchy 1’ and ‘Body Part - Hierarchy 1’, along with other features.

The following distribution was seen for the training data:

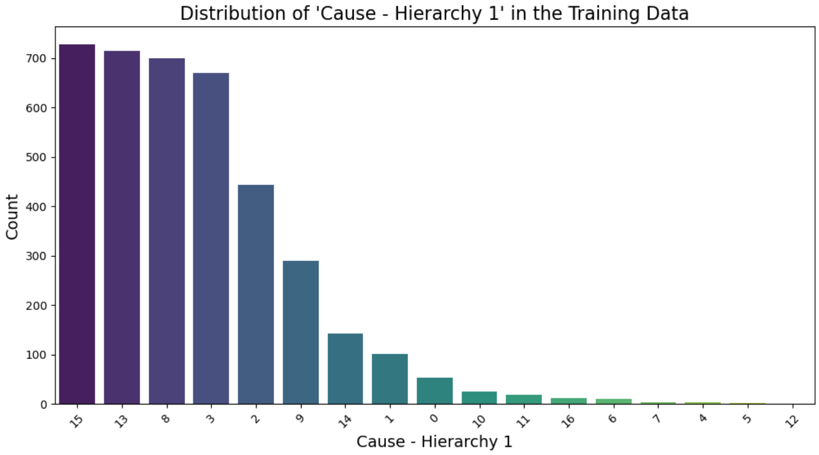


Figure 1

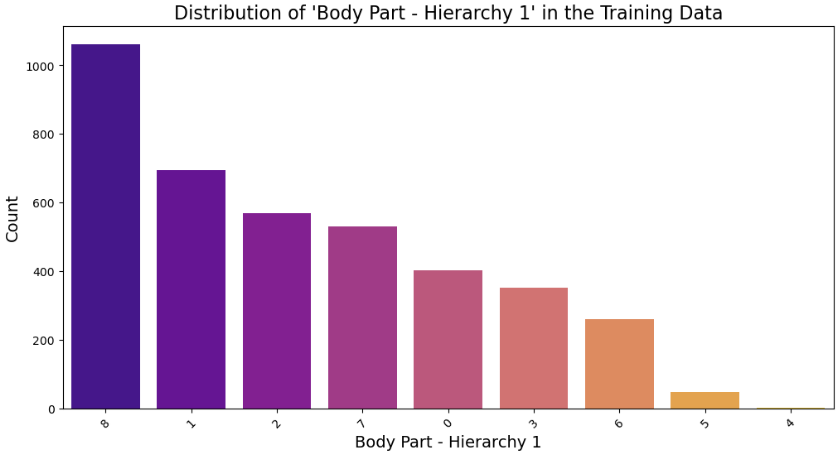


Figure 2

Figure 1 shows the distribution of the 'Cause - Hierarchy 1' variable in the training data. The x-axis represents the unique values or categories of 'Cause - Hierarchy 1', while the y-axis shows the count or frequency of each category.

From the bar chart, we can observe the following:

- The categories with the highest frequencies are 'Struck or Injured by', 'Strain or Injury by', 'Misc.', and 'Fall, Slip or Trip Injury', indicating that these are the most common causes of injuries or incidents in the dataset.
- Categories like 'Cut, Puncture, Scrape Injured by', 'Motor Vehicle', and 'Striking Against or Stepping on' also have relatively high frequencies, suggesting their significance as causes of incidents.
- Categories like 'Burn or Scald - Heat or Cold Exposures - Contact With', 'Rubbed or Abraded by', and other less frequent categories represent less common causes in the dataset.

This distribution provides insights into the most prevalent causes of injuries or incidents, which could help in identifying areas for preventive measures, safety protocols, or targeted training programs based on the dominant causal factors.

Figure 2 shows the distribution of the 'Body Part - Hierarchy 1' variable in the training data. The x-axis represents the unique values or categories of 'Body Part - Hierarchy 1', while the y-axis shows the count or frequency of each category.

From the bar chart, we can observe the following:

- The category with the highest frequency is 'Upper Extremities', which includes body parts like arms, hands, and fingers.
- The second highest category is 'Lower Extremities', which includes legs, feet, and toes.
- The 'Misc.' and 'Trunk' categories also have relatively high frequencies, indicating a significant number of injuries or incidents related to miscellaneous body parts and the torso/back region.
- Categories like 'Neck' and 'Multiple body parts' have lower frequencies, suggesting fewer cases involving these body parts.

This distribution provides insights into the most commonly affected body parts in the dataset, which could be useful for prioritizing safety measures or resource allocation based on the prevalent injury locations.

Overall, these visualizations offer a clear understanding of the data's characteristics, allowing stakeholders to prioritize their efforts based on the most frequently occurring body parts affected and the primary causes of incidents.