Data Preparation

In [33]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime as dt
```

In [34]:
```
df=pd.read_csv(r"C:/Users/User-PC/Desktop/Load_Csv/10 pipe dataset.csv")
df.head
```

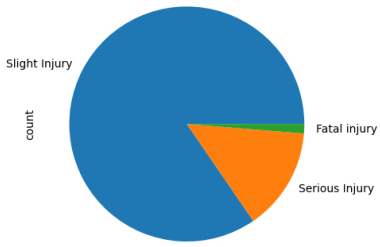
Out[34]:
```
<bound method NDFrame.head of           Time Day_of_week Age_band_of_driver Sex_of_driver  \
0      17:02:00      Monday              18-30          Male
1      17:02:00      Monday              31-50          Male
2      17:02:00      Monday              18-30          Male
3       1:06:00      Sunday              18-30          Male
4       1:06:00      Sunday              18-30          Male
...         ...         ...                ...           ...
12311  16:15:00   Wednesday              31-50          Male
12312  18:00:00      Sunday            Unknown          Male
12313  13:55:00      Sunday            Over 51          Male
12314  13:55:00      Sunday              18-30        Female
12315  13:55:00      Sunday              18-30          Male

        Educational_level Vehicle_driver_relation Driving_experience  \
0       Above high school              Employee              1-2yr
1      Junior high school              Employee          Above 10yr
2      Junior high school              Employee              1-2yr
3      Junior high school              Employee             5-10yr
4      Junior high school              Employee              2-5yr
...                   ...                   ...                ...
12311                 NaN              Employee              2-5yr
12312   Elementary school              Employee             5-10yr
12313  Junior high school              Employee             5-10yr
12314  Junior high school              Employee          Above 10yr
12315  Junior high school              Employee             5-10yr

           Type_of_vehicle Owner_of_vehicle Service_year_of_vehicle  ...  \
0              Automobile            Owner              Above 10yr  ...
1      Public (> 45 seats)            Owner                5-10yrs  ...
2         Lorry (41?100Q)            Owner                     NaN  ...
3      Public (> 45 seats)     Governmental                     NaN  ...
4                      NaN            Owner                5-10yrs  ...
...                    ...              ...                    ...  ...
12311       Lorry (11?40Q)            Owner                     NaN  ...
12312           Automobile            Owner                     NaN  ...
12313                Bajaj            Owner                2-5yrs  ...
12314      Lorry (41?100Q)            Owner                2-5yrs  ...
12315                Other            Owner                2-5yrs  ...

      Vehicle_movement   Casualty_class Sex_of_casualty Age_band_of_casualty  \
0       Going straight               na              na                   na
1       Going straight               na              na                   na
2       Going straight  Driver or rider            Male                31-50
3       Going straight       Pedestrian          Female                18-30
4       Going straight               na              na                   na
...                ...              ...             ...                  ...
12311   Going straight               na              na                   na
12312            Other               na              na                   na
12313            Other  Driver or rider            Male                31-50
12314            Other               na              na                   na
12315         Stopping       Pedestrian          Female                    5

      Casualty_severity Work_of_casuality Fitness_of_casuality  \
0                    na              NaN                  NaN
1                    na              NaN                  NaN
2                     3           Driver                  NaN
3                     3           Driver               Normal
4                    na              NaN                  NaN
...                 ...              ...                  ...
12311                na           Driver               Normal
12312                na           Driver               Normal
12313                 3           Driver               Normal
12314                na           Driver               Normal
12315                 3           Driver               Normal

                                 Pedestrian_movement  \
0                                Not a Pedestrian
1                                Not a Pedestrian
2                                Not a Pedestrian
3                                Not a Pedestrian
4                                Not a Pedestrian
...                                           ...
12311                            Not a Pedestrian
12312                            Not a Pedestrian
12313                            Not a Pedestrian
12314                            Not a Pedestrian
12315  Crossing from nearside - masked by parked or s...

                        Cause_of_accident Accident_severity
0                         Moving Backward     Slight Injury
1                               Overtaking     Slight Injury
2                  Changing lane to the left    Serious Injury
3                 Changing lane to the right     Slight Injury
4                               Overtaking     Slight Injury
...                                    ...               ...
12311                      No distancing     Slight Injury
12312                      No distancing     Slight Injury
12313             Changing lane to the right    Serious Injury
12314  Driving under the influence of drugs     Slight Injury
12315             Changing lane to the right     Slight Injury

[12316 rows x 32 columns]>
```

In [35]:
```
df.shape
```

Out[35]:
```
(12316, 32)
```

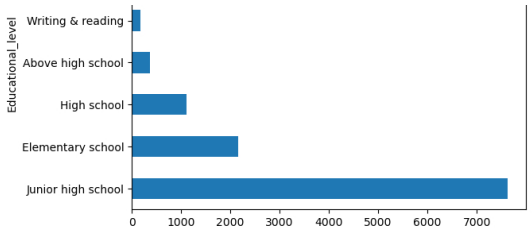
In [36]:
```
df['Accident_severity'].value_counts().plot(kind='pie')
```

Out[36]:
```
<Axes: ylabel='count'>
```



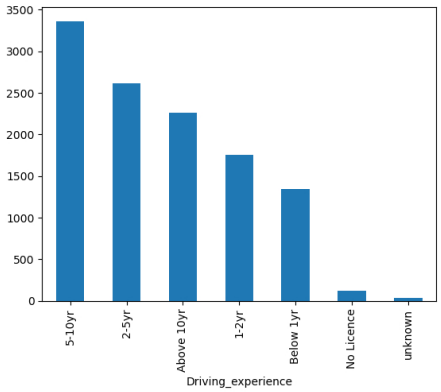
In [37]:
```
df['Educational_level'].value_counts().plot(kind='barh')
```

Out[37]:
```
<Axes: ylabel='Educational_level'>
```



In [38]:
```
df['Driving_experience'].value_counts().plot(kind='bar')
```

Out[38]:
```
<Axes: xlabel='Driving_experience'>
```



In [39]:
```
df['Time']=pd.to_datetime(df['Time'])
```

```
C:\Users\User-PC\AppData\Local\Temp\ipykernel_5864\3907310423.py:1: UserWarning: Could not infer format, so each element will be parsed individually, falling back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  df['Time']=pd.to_datetime(df['Time'])
```

In [40]:
```
df['Hour_of_day']= df['Time'].dt.hour
```

In [41]:
```
new_df=df.copy()
```

In [42]:
```
new_df.drop('Time',axis=1, inplace=True)
```

In [43]:
```
new_df
```

Out[43]:

	Day_of_week	Age_band_of_driver	Sex_of_driver	Educational_level	Vehicle_driver_relation	Driving_experience	Type_of_vehicle	Owner_of_vehicle	Service_year_of_vehicle	Defect_of_vehicle	...	Casualty_class	Sex_of_casualty	Age_band_of_casualty	Casualty_severity	Worl
0	Monday	18-30	Male	Above high school	Employee	1-2yr	Automobile	Owner	Above 10yr	No defect	...	na	na	na	na	
1	Monday	31-50	Male	Junior high school	Employee	Above 10yr	Public (> 45 seats)	Owner	5-10yrs	No defect	...	na	na	na	na	
2	Monday	18-30	Male	Junior high school	Employee	1-2yr	Lorry (41?100Q)	Owner	NaN	No defect	...	Driver or rider	Male	31-50	3	
3	Sunday	18-30	Male	Junior high school	Employee	5-10yr	Public (> 45 seats)	Governmental	NaN	No defect	...	Pedestrian	Female	18-30	3	
4	Sunday	18-30	Male	Junior high school	Employee	2-5yr	NaN	Owner	5-10yrs	No defect	...	na	na	na	na	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
12311	Wednesday	31-50	Male	NaN	Employee	2-5yr	Lorry (11?40Q)	Owner	NaN	No defect	...	na	na	na	na	
12312	Sunday	Unknown	Male	Elementary school	Employee	5-10yr	Automobile	Owner	NaN	No defect	...	na	na	na	na	
12313	Sunday	Over 51	Male	Junior high school	Employee	5-10yr	Bajaj	Owner	2-5yrs	No defect	...	Driver or rider	Male	31-50	3	
12314	Sunday	18-30	Female	Junior high school	Employee	Above 10yr	Lorry (41?100Q)	Owner	2-5yrs	No defect	...	na	na	na	na	
12315	Sunday	18-30	Male	Junior high school	Employee	5-10yr	Other	Owner	2-5yrs	No defect	...	Pedestrian	Female	5	3	

12316 rows × 32 columns

Encode Target Column

In [44]:
```
from sklearn.preprocessing import LabelEncoder
```

In [45]:
```
le=LabelEncoder()
new_df['Accident_severity']=le.fit_transform(new_df['Accident_severity'])
```

In [46]:
```
new_df['Accident_severity'].value_counts()
```

Out[46]:
```
Accident_severity
2    10415
1     1743
0      158
Name: count, dtype: int64
```

Data Balance

In [47]:
```
from imblearn.over_sampling import RandomOverSampler

X=new_df.drop(columns=['Accident_severity'],axis=1)
y=new_df['Accident_severity']
```

In [48]:
```
over_sampler=RandomOverSampler(random_state=42)

X_resampled, y_resampled= over_sampler.fit_resample(X,y)
```

In [49]:
```
y_resampled.value_counts()
```

Out[49]:
```
Accident_severity
2    10415
1    10415
0    10415
Name: count, dtype: int64
```

Train test Split

In [50]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X_resampled, y_resampled, test_size=0.2, random_state=42)
```

Fill Null Values

In [51]:
```
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
```

```
strategies = {
    3: 'most_frequent',   # Educational_level
    4: 'most_frequent',   # Vehicle_driver_relation
    5: 'most_frequent',   # Driving_experience
    6: 'most_frequent',   # Type_of_vehicle
    8: 'constant',        # Service_year_of_vehicle
    9: 'constant',        # Defect_of_vehicle
    10: 'most_frequent',  # Area_accident_occured
    11: 'most_frequent',  # Lanes_or_Medians
    12: 'most_frequent',  # Road_allignment
    13: 'most_frequent',  # Types_of_Junction
    14: 'most_frequent',  # Road_surface_type
    18: 'most_frequent',  # Type_of_collision
    21: 'most_frequent',  # Vehicle_movement
    26: 'most_frequent',  # Work_of_casuality
    27: 'most_frequent'   # Fitness_of_casuality
}

tf1 = ColumnTransformer([
    ('impute_educational_level', SimpleImputer(strategy=strategies[3]), [3]),
    ('impute_Vehicle_driver_relation', SimpleImputer(strategy=strategies[4]), [4]),
    ('impute_Driving_experience', SimpleImputer(strategy=strategies[5]), [5]),
    ('impute_Type_of_vehicle', SimpleImputer(strategy=strategies[6]), [6]),
    ('impute_Service_year_of_vehicle', SimpleImputer(strategy=strategies[8], fill_value='Unknown'), [8]),
    ('impute_Defect_of_vehicle', SimpleImputer(strategy=strategies[9], fill_value='Unknown'), [9]),
    ('impute_Area_accident_occured', SimpleImputer(strategy=strategies[10]), [10]),
    ('impute_Lanes_or_Medians', SimpleImputer(strategy=strategies[11]), [11]),
    ('impute_Road_allignment', SimpleImputer(strategy=strategies[12]), [12]),
    ('impute_Types_of_Junction', SimpleImputer(strategy=strategies[13]), [13]),
    ('impute_Road_surface_type', SimpleImputer(strategy=strategies[14]), [14]),
    ('impute_Type_of_collision', SimpleImputer(strategy=strategies[18]), [18]),
    ('impute_Vehicle_movement', SimpleImputer(strategy=strategies[21]), [21]),
    ('impute_Work_of_casuality', SimpleImputer(strategy=strategies[26]), [26]),
    ('impute_Fitness_of_casuality', SimpleImputer(strategy=strategies[27]), [27])
], remainder='passthrough')

new_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12316 entries, 0 to 12315
Data columns (total 32 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Day_of_week                  12316 non-null  object
 1   Age_band_of_driver           12316 non-null  object
 2   Sex_of_driver                12316 non-null  object
 3   Educational_level            11575 non-null  object
 4   Vehicle_driver_relation      11737 non-null  object
 5   Driving_experience           11487 non-null  object
 6   Type_of_vehicle              11366 non-null  object
 7   Owner_of_vehicle             11834 non-null  object
 8   Service_year_of_vehicle      8388 non-null   object
 9   Defect_of_vehicle            7889 non-null   object
 10  Area_accident_occured        12077 non-null  object
 11  Lanes_or_Medians             11931 non-null  object
 12  Road_allignment              12174 non-null  object
 13  Types_of_Junction            11429 non-null  object
 14  Road_surface_type            12144 non-null  object
 15  Road_surface_conditions      12316 non-null  object
 16  Light_conditions             12316 non-null  object
 17  Weather_conditions           12316 non-null  object
 18  Type_of_collision            12161 non-null  object
 19  Number_of_vehicles_involved  12316 non-null  int64
 20  Number_of_casualties         12316 non-null  int64
 21  Vehicle_movement             12008 non-null  object
 22  Casualty_class               12316 non-null  object
 23  Sex_of_casualty              12316 non-null  object
 24  Age_band_of_casualty         12316 non-null  object
 25  Casualty_severity            12316 non-null  object
 26  Work_of_casuality            9118 non-null   object
 27  Fitness_of_casuality         9681 non-null   object
 28  Pedestrian_movement          12316 non-null  object
 29  Cause_of_accident            12316 non-null  object
 30  Accident_severity            12316 non-null  int32
 31  Hour_of_day                  12316 non-null  int32
dtypes: int32(2), int64(2), object(28)
memory usage: 2.9+ MB
```

Encode Categorical Columns

In [52]:
```
from sklearn.preprocessing import OneHotEncoder
object_column_indices=[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30]

tf2=ColumnTransformer(
    [
        (f"ohe_{col}",OneHotEncoder(sparse_output=False,handle_unknown='ignore'),[col])
        for col in object_column_indices
    ], remainder='passthrough'
)
```

In [53]:
```
from sklearn.preprocessing import MinMaxScaler

tf3=ColumnTransformer(
    [
        ("min_max_scaling",MinMaxScaler(),slice(0,30))
], remainder='passthrough'
)
```

Feature Selection using 'Chi2' Statistics

chi2: This is one of the scoring functions available for feature selection in scikit-learn. It calculates the chi-squared statistic between each feature and the target variable (accidents) to determine the relevance of each feature. chi2 is commonly used for feature selection when dealing with categorical target variables.

In [54]:
```
from sklearn.feature_selection import SelectKBest, chi2
tf4= SelectKBest(chi2, k=50)
```

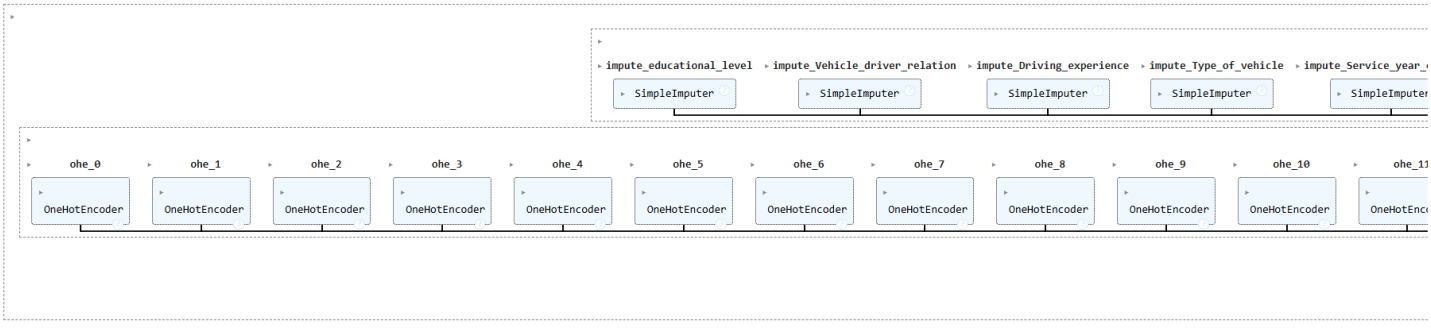
Model (Random Forest Classifier)

In [55]:
```
from sklearn.ensemble import RandomForestClassifier
tf5= RandomForestClassifier()
```

Create Pipeline

In [56]:
```
from sklearn.pipeline import Pipeline

pipe= Pipeline([
    ('trf1',tf1),
    ('trf2',tf2),
    ('trf4',tf4),
    ('trf5',tf5)
])
pipe.fit(X_train,y_train)
```

Out[56]:



Explore the Pipeline

In [57]:
```
pipe.feature_names_in_
pipe.named_steps
```

Out[57]:
```
{'trf1': ColumnTransformer(remainder='passthrough',
                   transformers=[('impute_educational_level',
                                  SimpleImputer(strategy='most_frequent'), [3]),
                                 ('impute_Vehicle_driver_relation',
                                  SimpleImputer(strategy='most_frequent'), [4]),
                                 ('impute_Driving_experience',
                                  SimpleImputer(strategy='most_frequent'), [5]),
                                 ('impute_Type_of_vehicle',
                                  SimpleImputer(strategy='most_frequent'), [6...
                                 ('impute_Road_surface_type',
                                  SimpleImputer(strategy='most_frequent'),
                                  [14]),
                                 ('impute_Type_of_collision',
                                  SimpleImputer(strategy='most_frequent'),
                                  [18]),
                                 ('impute_Vehicle_movement',
                                  SimpleImputer(strategy='most_frequent'),
                                  [21]),
                                 ('impute_Work_of_casuality',
                                  SimpleImputer(strategy='most_frequent'),
                                  [26]),
                                 ('impute_Fitness_of_casuality',
                                  SimpleImputer(strategy='most_frequent'),
                                  [27])])),
 'trf2': ColumnTransformer(remainder='passthrough',
                   transformers=[('ohe_0',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [0]),
                                 ('ohe_1',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [1]),
                                 ('ohe_2',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [2]),
                                 ('ohe_3',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [3]),
                                 ('...
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [25]),
                                 ('ohe_26',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [26]),
                                 ('ohe_27',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [27]),
                                 ('ohe_28',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [28]),
                                 ('ohe_29',
                                  OneHotEncoder(handle_unknown='ignore',
                                                sparse_output=False),
                                  [29]), ...]),
 'trf4': SelectKBest(k=50, score_func=<function chi2 at 0x000001F66BCDF100>),
 'trf5': RandomForestClassifier()}
```

Accuracy Score

In [58]:
```
from sklearn.metrics import accuracy_score

y_pred=pipe.predict(X_test)
accuracy_score(y_test,y_pred)
```

Out[58]:
```
0.9249479916786686
```

Classification Report

In [59]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      2085
           1       0.86      0.95      0.90      2100
           2       0.95      0.83      0.88      2064

    accuracy                           0.92      6249
   macro avg       0.93      0.92      0.92      6249
weighted avg       0.93      0.92      0.92      6249
```

Confusion Matrix

In [60]:
```
from sklearn.metrics import confusion_matrix
print(confusion_matrix
      (y_test,y_pred))
```

```
[[2085    0    0]
 [  22 1991   87]
 [  29  331 1704]]
```

Save Pipeline

In [61]:
```
import pickle
pickle.dump(pipe,open('pipe.pkl','wb'))
```

Prediction System

In [62]:
```
import numpy as np
```

In [63]:
```
def pred(Day_of_week, Age_band_of_driver, Sex_of_driver, Educational_level, Vehicle_driver_relation,
         Driving_experience, Type_of_vehicle, Owner_of_vehicle, Service_year_of_vehicle,
         Defect_of_vehicle, Area_accident_occured, Lanes_or_Medians, Road_allignment,
         Types_of_Junction, Road_surface_type, Road_surface_conditions, Light_conditions,
         Weather_conditions, Type_of_collision, Number_of_vehicles_involved,
         Number_of_casualties, Vehicle_movement, Casualty_class, Sex_of_casualty,
         Age_band_of_casualty, Casualty_severity, Work_of_casuality, Fitness_of_casuality,
         Pedestrian_movement, Cause_of_accident, Hour_of_Day):

    features=np.array([[Day_of_week, Age_band_of_driver, Sex_of_driver, Educational_level, Vehicle_driver_relation,
         Driving_experience, Type_of_vehicle, Owner_of_vehicle, Service_year_of_vehicle,
         Defect_of_vehicle, Area_accident_occured, Lanes_or_Medians, Road_allignment,
         Types_of_Junction, Road_surface_type, Road_surface_conditions, Light_conditions,
         Weather_conditions, Type_of_collision, Number_of_vehicles_involved,
         Number_of_casualties, Vehicle_movement, Casualty_class, Sex_of_casualty,
         Age_band_of_casualty, Casualty_severity, Work_of_casuality, Fitness_of_casuality,
         Pedestrian_movement, Cause_of_accident, Hour_of_Day
    ]])

    results=pipe.predict(features)
    return results
```

In [64]:
```
predicted_class = pred(Day_of_week="Thursday",
                       Age_band_of_driver='31-50',
                       Sex_of_driver='Male',
                       Educational_level='Junior high school',
                       Vehicle_driver_relation='Owner',
                       Driving_experience=None,
                       Type_of_vehicle='Long lorry',
                       Owner_of_vehicle='Owner',
                       Service_year_of_vehicle='Unknown'
```

```python
                                 Service_year_of_vehicle='Unknown',
                                 Defect_of_vehicle=None,
                                 Area_accident_occured='Other',
                                 Lanes_or_Medians='Two-way (divided with solid lines road marking)',
                                 Road_allignment='Tangent road with flat terrain',
                                 Types_of_Junction=None,
                                 Road_surface_type=None,
                                 Road_surface_conditions='Dry',
                                 Light_conditions='Daylight',
                                 Weather_conditions='Normal',
                                 Type_of_collision='Collision with animals',
                                 Number_of_vehicles_involved=2,
                                 Number_of_casualties=1,
                                 Vehicle_movement='Going straight',
                                 Casualty_class='Driver or rider',
                                 Sex_of_casualty='Male',
                                 Age_band_of_casualty='18-30',
                                 Casualty_severity=3,
                                 Work_of_casuality='Driver',
                                 Fitness_of_casuality='Normal',
                                 Pedestrian_movement='Not a Pedestrian',
                                 Cause_of_accident='Changing lane to the left',
                                 Hour_of_Day=12)

if predicted_class[0] == 2:
    print("Slight Injury.....")
elif predicted_class[0] == 1:
    print("Serious Injury")
else:
    print("Fatal Injury")
```

```
Serious Injury
```

```
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
C:\Users\User-PC\anaconda3\Lib\site-packages\sklearn\base.py:493: UserWarning: X does not have valid feature names, but SimpleImputer was fitted with feature names
  warnings.warn(
```

In [ ]: