Load Data

In [12]:
```
!pip install kagglehub
import kagglehub
```

```
Requirement already satisfied: kagglehub in c:\users\alana\onedrive\desktop\ml\footb
all injury project\football-env\lib\site-packages (0.3.13)
Requirement already satisfied: packaging in c:\users\alana\onedrive\desktop\ml\footb
all injury project\football-env\lib\site-packages (from kagglehub) (25.0)
Requirement already satisfied: pyyaml in c:\users\alana\onedrive\desktop\ml\football
injury project\football-env\lib\site-packages (from kagglehub) (6.0.2)
Requirement already satisfied: requests in c:\users\alana\onedrive\desktop\ml\footba
ll injury project\football-env\lib\site-packages (from kagglehub) (2.32.5)
Requirement already satisfied: tqdm in c:\users\alana\onedrive\desktop\ml\football i
njury project\football-env\lib\site-packages (from kagglehub) (4.67.1)
Requirement already satisfied: charset_normalizer<4,>=2 in c:\users\alana\onedrive\d
esktop\ml\football injury project\football-env\lib\site-packages (from requests->kag
glehub) (3.4.3)
Requirement already satisfied: idna<4,>=2.5 in c:\users\alana\onedrive\desktop\ml\fo
otball injury project\football-env\lib\site-packages (from requests->kagglehub) (3.1
0)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\alana\onedrive\desktop
\ml\football injury project\football-env\lib\site-packages (from requests->kagglehu
b) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\alana\onedrive\desktop
\ml\football injury project\football-env\lib\site-packages (from requests->kagglehu
b) (2025.8.3)
Requirement already satisfied: colorama in c:\users\alana\onedrive\desktop\ml\footba
ll injury project\football-env\lib\site-packages (from tqdm->kagglehub) (0.4.6)
```

In [13]:
```
path = kagglehub.dataset_download("yuanchunhong/university-football-injury-predicti
```

In [14]:
```
import os
import pandas as pd
print(("files in directory:", os.listdir(path)))
```

```
('files in directory:', ['data.csv'])
```

In [15]:
```
csv_file = os.path.join(path, "data.csv")
data = pd.read_csv(csv_file)
```

In [16]:
```
data.head(10)
```

Out[16]:

	Age	Height_cm	Weight_kg	Position	Training_Hours_Per_Week	Matches_Played_Past
0	22	173	64	Midfielder	11.575308	
1	18	170	67	Midfielder	12.275869	
2	22	186	75	Forward	12.254896	
3	20	172	62	Defender	9.006678	
4	18	172	94	Midfielder	12.683668	
5	23	189	89	Goalkeeper	10.262987	
6	22	189	71	Midfielder	8.069288	
7	23	184	75	Goalkeeper	6.407939	
8	22	174	71	Midfielder	10.554931	
9	23	185	76	Midfielder	11.899732	

In [17]:
```
data.shape
```

Out[17]:
```
(800, 19)
```

In [18]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 800 entries, 0 to 799
Data columns (total 19 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   Age                        800 non-null    int64  
 1   Height_cm                  800 non-null    int64  
 2   Weight_kg                  800 non-null    int64  
 3   Position                   800 non-null    object 
 4   Training_Hours_Per_Week    800 non-null    float64
 5   Matches_Played_Past_Season 800 non-null    int64  
 6   Previous_Injury_Count      800 non-null    int64  
 7   Knee_Strength_Score        800 non-null    float64
 8   Hamstring_Flexibility      800 non-null    float64
 9   Reaction_Time_ms           800 non-null    float64
 10  Balance_Test_Score         800 non-null    float64
 11  Sprint_Speed_10m_s         800 non-null    float64
 12  Agility_Score              800 non-null    float64
 13  Sleep_Hours_Per_Night      800 non-null    float64
 14  Stress_Level_Score         800 non-null    float64
 15  Nutrition_Quality_Score    800 non-null    float64
 16  Warmup_Routine_Adherence   800 non-null    int64  
 17  Injury_Next_Season         800 non-null    int64  
 18  BMI                        800 non-null    float64
dtypes: float64(11), int64(7), object(1)
memory usage: 118.9+ KB
```

Data Preprocessing

In [19]:
```
data['Position'] = data['Position'].astype('category')
```

In [20]:
```
data.describe().T
```

Out[20]:

	count	mean	std	min	25%	5
Age	800.0	21.135000	1.991037	18.000000	19.000000	21.000
Height_cm	800.0	177.407500	7.148974	154.000000	173.000000	177.000
Weight_kg	800.0	73.235000	9.929276	45.000000	66.000000	73.000
Training_Hours_Per_Week	800.0	9.951150	2.610395	5.000000	8.127151	9.895
Matches_Played_Past_Season	800.0	22.332500	10.311516	5.000000	13.000000	22.000
Previous_Injury_Count	800.0	1.536250	1.292584	0.000000	1.000000	1.000
Knee_Strength_Score	800.0	74.933249	6.672704	52.391351	70.432656	74.997
Hamstring_Flexibility	800.0	79.154123	6.782332	58.180381	74.495959	79.187
Reaction_Time_ms	800.0	249.423244	22.532387	180.000000	234.089585	249.127
Balance_Test_Score	800.0	83.832337	6.931657	60.059484	79.044910	84.156
Sprint_Speed_10m_s	800.0	5.949025	0.329133	4.862435	5.732552	5.937
Agility_Score	800.0	78.341311	8.775418	50.000000	72.675392	78.340
Sleep_Hours_Per_Night	800.0	7.417124	0.793183	5.000000	6.850062	7.424
Stress_Level_Score	800.0	54.039342	11.421143	21.561186	45.775371	54.047
Nutrition_Quality_Score	800.0	74.382174	9.324899	50.000000	67.809084	74.363
Warmup_Routine_Adherence	800.0	0.597500	0.490708	0.000000	0.000000	1.000
Injury_Next_Season	800.0	0.500000	0.500313	0.000000	0.000000	0.500
BMI	800.0	23.377364	3.673279	14.346326	20.786644	23.130

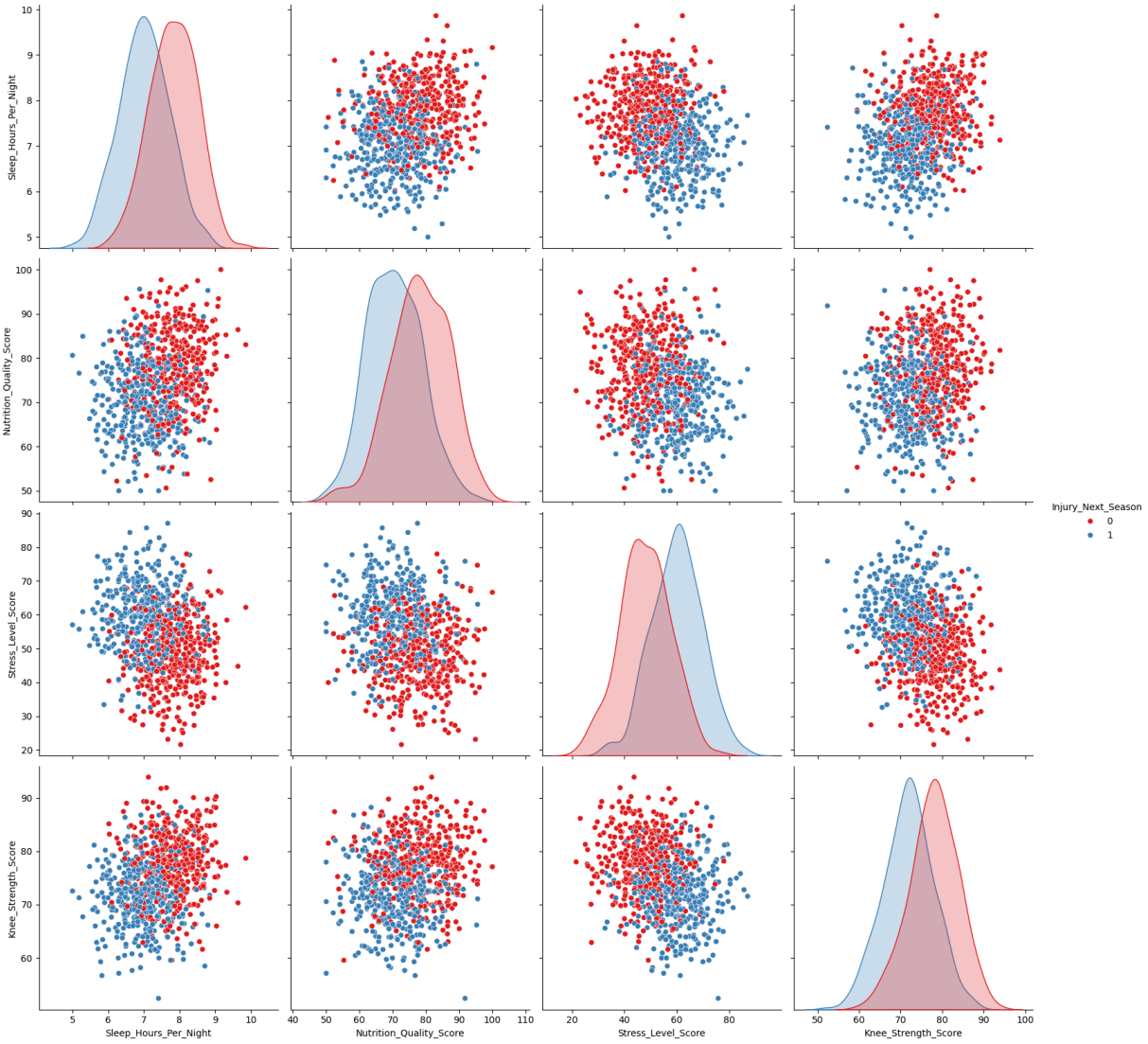
In [21]:
```
injury_data = data.groupby("Injury_Next_Season").mean(numeric_only=True).round(2)
injury_data
```

Out[21]:

Injury_Next_Season	Age	Height_cm	Weight_kg	Training_Hours_Per_Week	Matches_Playe
0	21.09	177.80	73.03	9.86	
1	21.18	177.02	73.44	10.04	

In [22]:
```
import seaborn as sns
sns.pairplot(
    data[["Sleep_Hours_Per_Night", "Nutrition_Quality_Score", "Stress_Level_Score",
    hue = "Injury_Next_Season",
    height = 4,
    palette = "Set1"
)
```

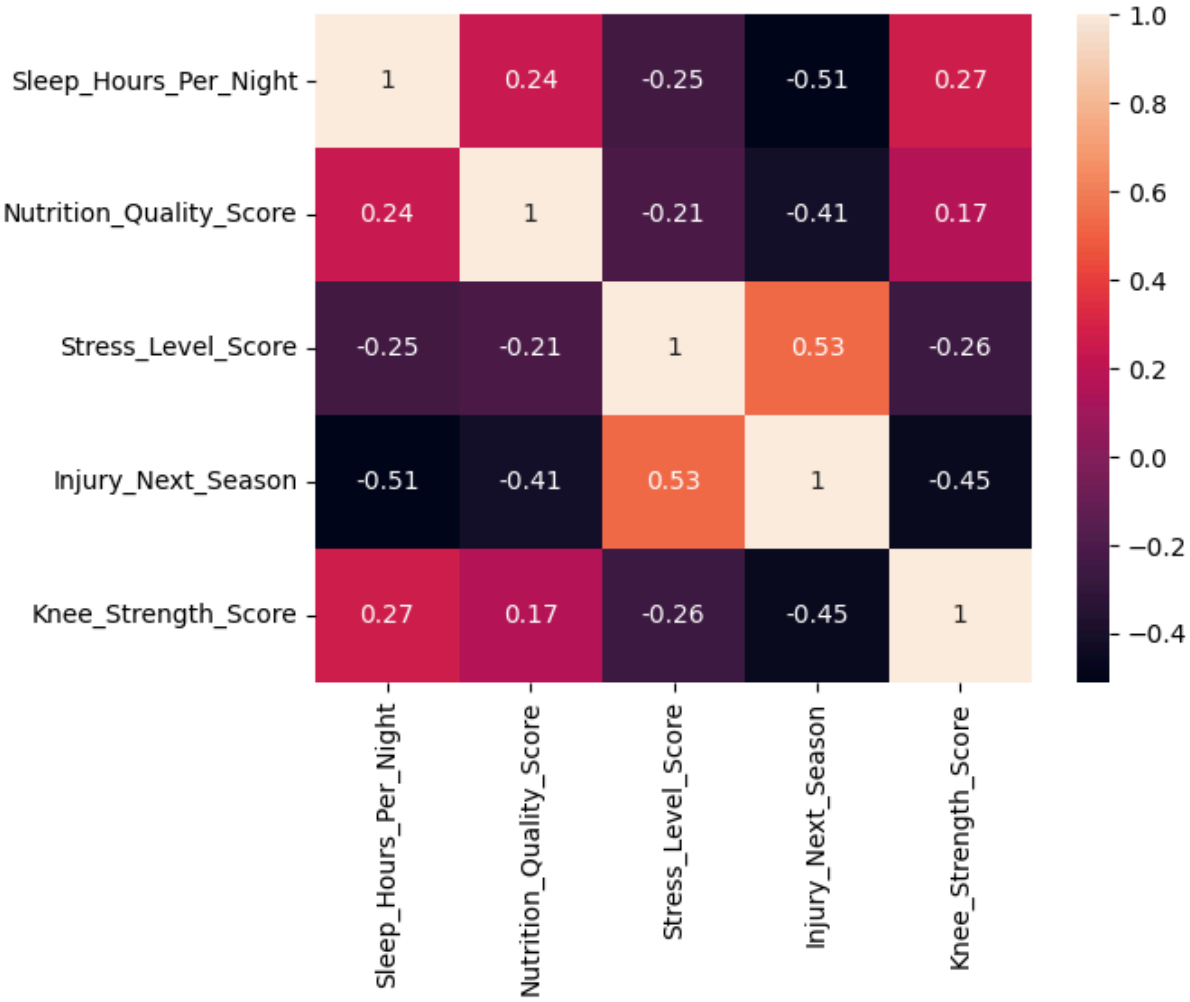
Out[22]: <seaborn.axisgrid.PairGrid at 0x26931357bc0>



In [23]:
```
sns.heatmap(
    data[["Sleep_Hours_Per_Night", "Nutrition_Quality_Score", "Stress_Level_Score",
```

```
    annot= True)
```

Out[23]: <Axes: >



In [24]:
```
data = pd.get_dummies(data)
```

Logistic Regression Model Build

In [25]:
```
data.columns
```

Out[25]:
```
Index(['Age', 'Height_cm', 'Weight_kg', 'Training_Hours_Per_Week',
       'Matches_Played_Past_Season', 'Previous_Injury_Count',
       'Knee_Strength_Score', 'Hamstring_Flexibility', 'Reaction_Time_ms',
       'Balance_Test_Score', 'Sprint_Speed_10m_s', 'Agility_Score',
       'Sleep_Hours_Per_Night', 'Stress_Level_Score',
       'Nutrition_Quality_Score', 'Warmup_Routine_Adherence',
       'Injury_Next_Season', 'BMI', 'Position_Defender', 'Position_Forward',
       'Position_Goalkeeper', 'Position_Midfielder'],
      dtype='object')
```

In [26]:
```
y = data["Injury_Next_Season"]
```

In [27]:
```
x = data.drop("Injury_Next_Season", axis = 1)
```

In [28]:
```
from sklearn.model_selection import train_test_split
```

In [29]:
```
x_train, x_test, y_train, y_test = train_test_split(
    x, y ,
    train_size = 0.8,
    random_state = 1
)
```

In [30]:
```
from sklearn.linear_model import LogisticRegression
```

In [31]:
```
lr = LogisticRegression(penalty=None, solver = 'newton-cg', max_iter = 150)
```

In [32]:
```
lr.fit(x_train, y_train)
```

Out[32]:

▾ LogisticRegression

▶ Parameters

Model accuracy score

In [33]:
```
round(lr.score(x_train, y_train), 3)
```

Out[33]: 0.964

In [34]:
```
round(lr.score(x_test, y_test), 3)
```

Out[34]: 0.931

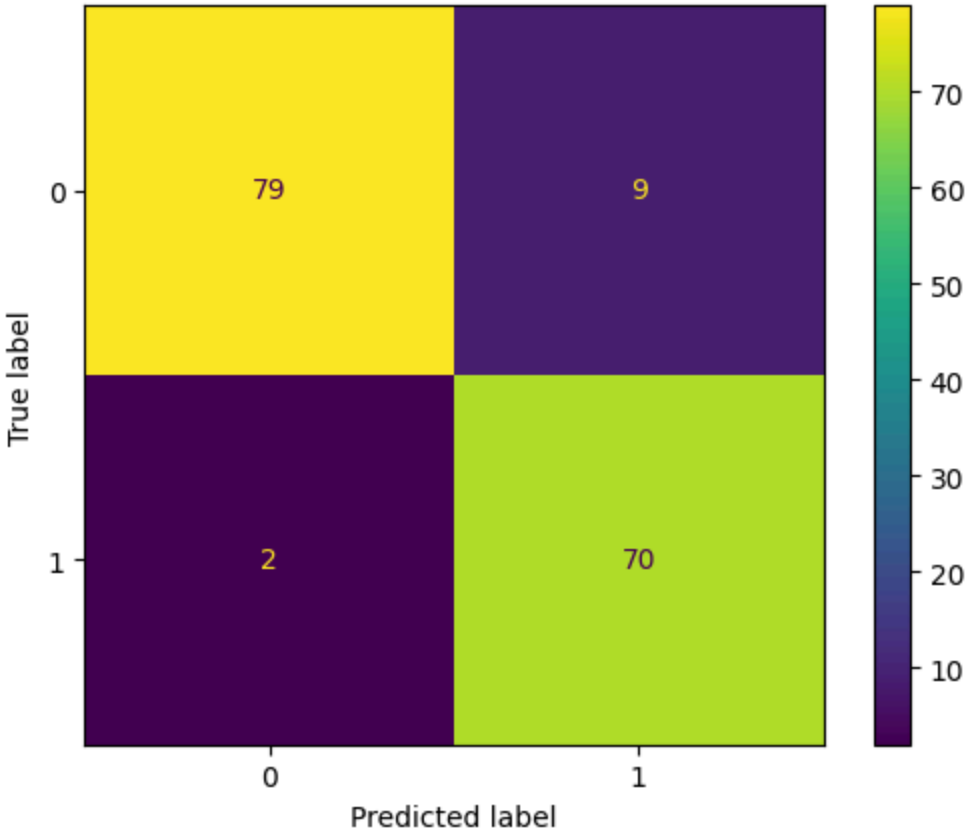
Confusion Matrix

In [35]:
```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

pred = lr.predict(x_test)
ypred = list(map(round, pred))
cm = confusion_matrix(y_test, ypred)

disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
```

Out[35]: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x26938f2eea0>



Model Prediction

In [36]:
```
data_new = x_train[:5]
```

In [37]:
```
lr.predict(data_new)
```

Out[37]:
```
array([1, 0, 0, 1, 0])
```

In [38]:
```
y_train[:5]
```

Out[38]:
```
797    1
411    0
0      0
318    1
555    0
Name: Injury_Next_Season, dtype: int64
```

Save for deployment

In [39]:
```
import pickle
import os
```

In [40]:
```
os.makedirs('models', exist_ok=True)
with open('models/footballmod.pkl', 'wb') as f:
```

```
    pickle.dump(lr, f)
```

In [41]:
```
x_train[:2]
```

Out[41]:

	Age	Height_cm	Weight_kg	Training_Hours_Per_Week	Matches_Played_Past_Season	F
797	24	182	75	5.494318	17	
411	20	188	80	7.540744	5	

2 rows × 21 columns