========================================================================

### **Hyperparameters Used for Readmission Prediction**

========================================================================

#### **Binary Classifiers**

1. **DecisionTreeClassifier**:
	* random\_state: 42
	* No specific hyperparameters were mentioned in the code, custom class weights were applied.
2. **LogisticRegression**:
	* random\_state: 42
	* max\_iter: 1000
	* class\_weight: 'balanced'
3. **RandomForestClassifier**:
	* random\_state: 42
	* class\_weight: 'balanced'
	* n\_jobs: -1 (in custom weighting)
4. **GradientBoostingClassifier**:
	* n\_estimators: 100
	* learning\_rate: 0.1 (in custom weighting)
	* random\_state: 42
	* sample\_weight: computed using custom weighting
5. **XGBClassifier**:
	* random\_state: 42
	* ratio of negative to positive class (in custom weighting): scale\_pos\_weight=float(counts[0] / counts[1],random\_state=42
6. **MLPClassifier**:
	* random\_state: 42
	* max\_iter: 1000
	* learning\_rate: 'adaptive' (in version 2 and custom weighting)

===================================================

#### **B. Multi-Label Classifiers**

1. **DecisionTreeClassifier**:
	* random\_state: 42
	* class\_weight: 'balanced'
2. **LogisticRegression**:
	* random\_state: 42
	* max\_iter: 1000
	* class\_weight: 'balanced'
3. **RandomForestClassifier**:
	* random\_state: 42
	* class\_weight: 'balanced'
	* n\_jobs: -1 (in custom weighting)
4. **GradientBoostingClassifier**:
	* n\_estimators: 100
	* learning\_rate: 0.1
	* random\_state: 42
	* sample\_weight: computed using compute\_sample\_weight(class\_weight='balanced', y=y\_train) in version 2
5. **XGBClassifier**:
	* random\_state: 42
	* sample\_weight: computed using compute\_sample\_weight(class\_weight='balanced', y=y\_train) in version 2
	* scale\_pos\_weight: ratio of negative to positive class (in custom weighting)
6. **MLPClassifier**:
	* random\_state: 42
	* max\_iter: 1000
	* learning\_rate: 'adaptive'