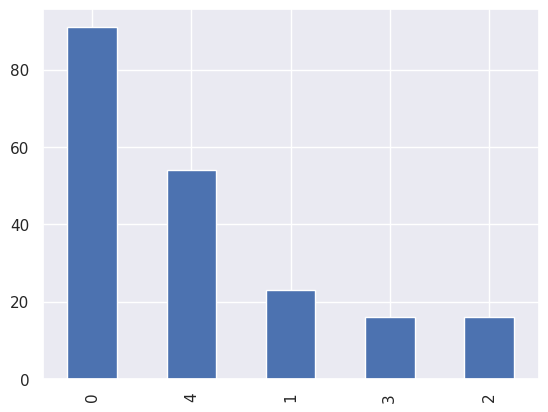
**OUTPUT**

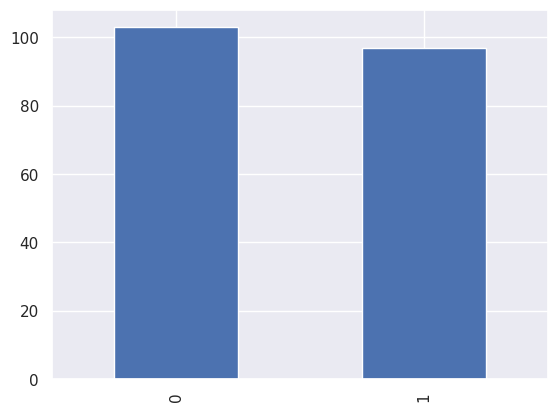
|  | **Age** | **Sex** | **BP** | **Cholesterol** | **Na\_to\_K** | **Drug** |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | 23 | F | HIGH | HIGH | 25.355 | DrugY |
| **1** | 47 | M | LOW | HIGH | 13.093 | drugC |
| **2** | 47 | M | LOW | HIGH | 10.114 | drugC |
| **3** | 28 | F | NORMAL | HIGH | 7.798 | drugX |
| **4** | 61 | F | LOW | HIGH | 18.043 | DrugY |

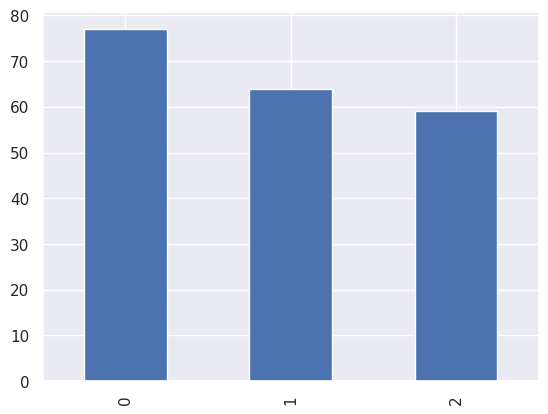
* array(['drugY', 'drugC', 'drugX', 'drugA', 'drugB'], dtype=object)
* 5
* array(['HIGH', 'NORMAL'], dtype=object)
* array(['HIGH', 'LOW', 'NORMAL'], dtype=object)
* array(['F', 'M'], dtype=object)

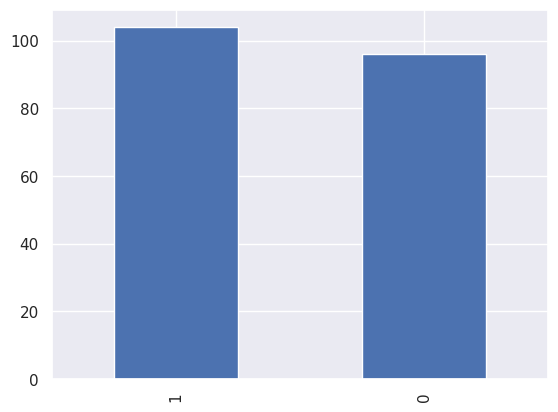
|  | **Age** | **Sex** | **BP** | **Cholesterol** | **Na\_to\_K** | **Drug** |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | 23 | 0 | 0 | 0 | 25.355 | 0 |
| **1** | 47 | 1 | 1 | 0 | 13.093 | 3 |
| **2** | 47 | 1 | 1 | 0 | 10.114 | 3 |
| **3** | 28 | 0 | 2 | 0 | 7.798 | 4 |
| **4** | 61 | 0 | 1 | 0 | 18.043 | 0 |

* **200**

****

****

****

****

* <class 'pandas.core.frame.DataFrame'>

RangeIndex: 200 entries, 0 to 199

Data columns (total 6 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Age 200 non-null int64

1 Sex 200 non-null int64

2 BP 200 non-null int64

3 Cholesterol 200 non-null int64

4 Na\_to\_K 200 non-null float64

5 Drug 200 non-null int64

dtypes: float64(1), int64(5)

memory usage: 9.5 KB

Age 0

Sex 0

BP 0

Cholesterol 0

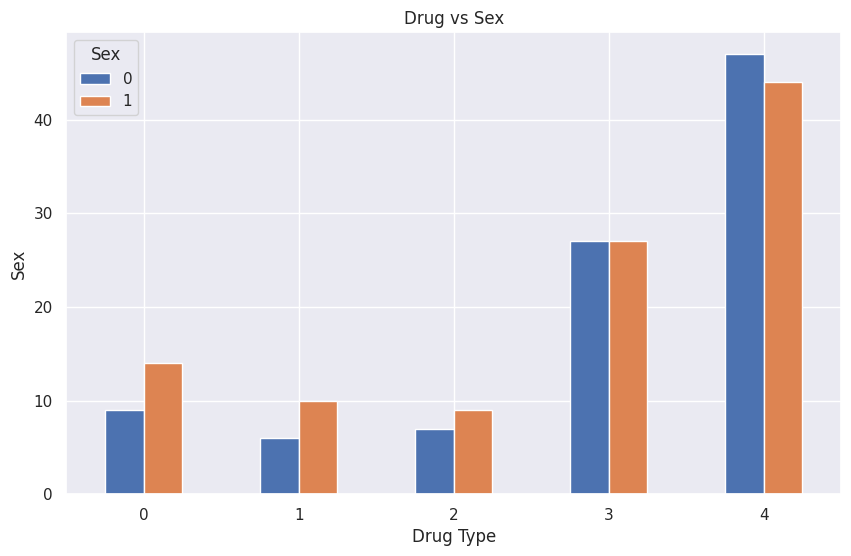
Na\_to\_K 0

Drug 0

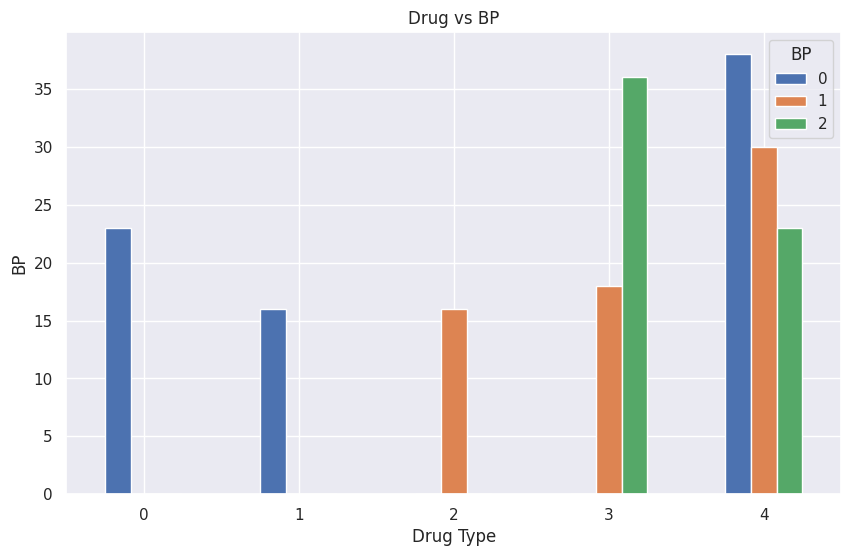
dtype: int64

|  | **Age** | **Sex** | **BP** | **Cholesterol** | **Na\_to\_K** | **Drug** |
| --- | --- | --- | --- | --- | --- | --- |
| **count** | 200.000000 | 200.000000 | 200.000000 | 200.000000 | 200.000000 | 200.000000 |
| **mean** | 44.315000 | 0.520000 | 0.910000 | 0.485000 | 16.084485 | 2.870000 |
| **std** | 16.544315 | 0.500854 | 0.821752 | 0.501029 | 7.223956 | 1.372047 |
| **min** | 15.000000 | 0.000000 | 0.000000 | 0.000000 | 6.269000 | 0.000000 |
| **25%** | 31.000000 | 0.000000 | 0.000000 | 0.000000 | 10.445500 | 2.000000 |
| **50%** | 45.000000 | 1.000000 | 1.000000 | 0.000000 | 13.936500 | 3.000000 |
| **75%** | 58.000000 | 1.000000 | 2.000000 | 1.000000 | 19.380000 | 4.000000 |
| **max** | 74.000000 | 1.000000 | 2.000000 | 1.000000 | 38.247000 | 4.000000 |

| **Sex** | **0** | **1** |
| --- | --- | --- |
| **Drug** |  |  |
| **0** | 9 | 14 |
| **1** | 6 | 10 |
| **2** | 7 | 9 |
| **3** | 27 | 27 |
| **4** | 47 | 44 |

****

| **BP** | **0** | **1** | **2** |
| --- | --- | --- | --- |
| **Drug** |  |  |  |
| **0** | 23 | 0 | 0 |
| **1** | 16 | 0 | 0 |
| **2** | 0 | 16 | 0 |
| **3** | 0 | 18 | 36 |
| **4** | 38 | 30 | 23 |

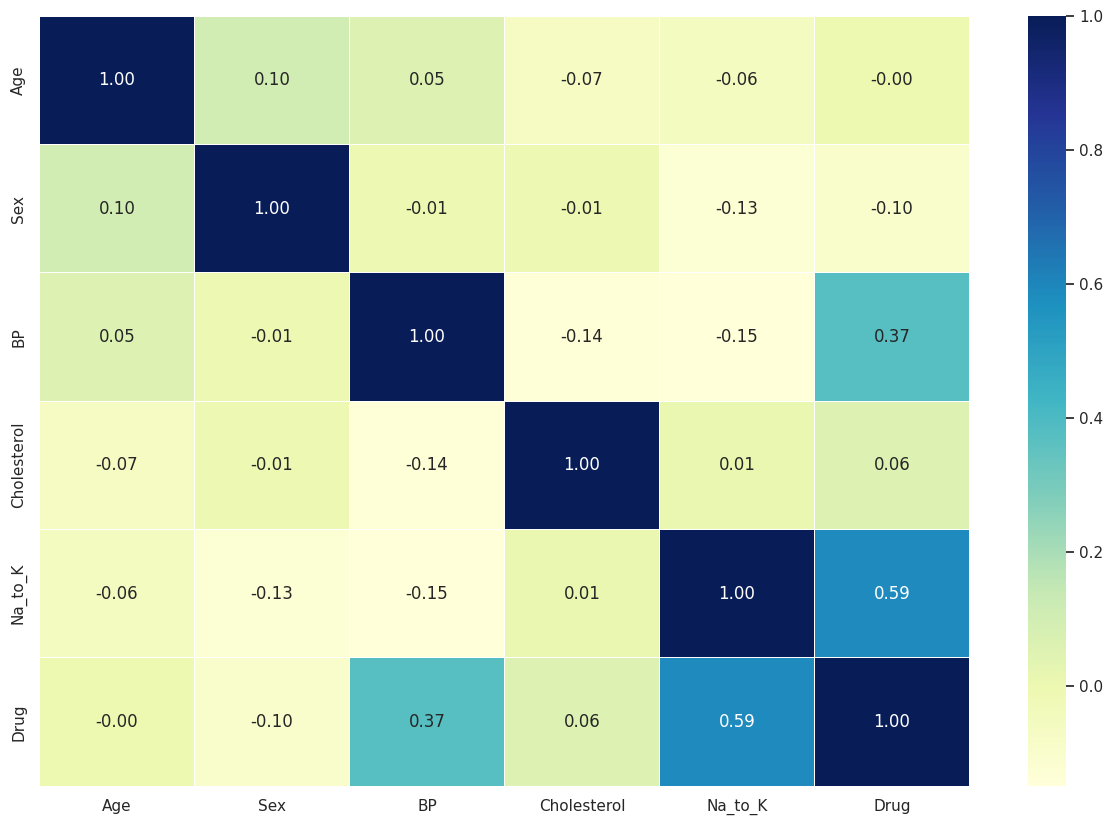
****

| **Drug** |  |  |
| --- | --- | --- |
| **0** | 12 | 11 |
| **1** | 8 | 8 |
| **2** | 16 | 0 |
| **3** | 20 | 34 |
| **4** | 47 | 44 |

| **Cholesterol** | **0** | **1** |
| --- | --- | --- |
| **Sex** | | | |  |  |
| **0** | | | | 49 | 47 |
| **1** | | | | 54 | 50 |

| **BP** | **0** | **1** | **2** |
| --- | --- | --- | --- |
| **Sex** |  |  |  |
| **0** | 38 | 28 | 30 |
| **1** | 39 | 36 | 29 |

| **BP** | **0** | **1** | **2** |
| --- | --- | --- | --- |
| **Cholesterol** |  |  |  |
| **0** | 35 | 31 | 37 |
| **1** | 42 | 33 | 22 |

****

* **5**
* [{'Age': 0.946},

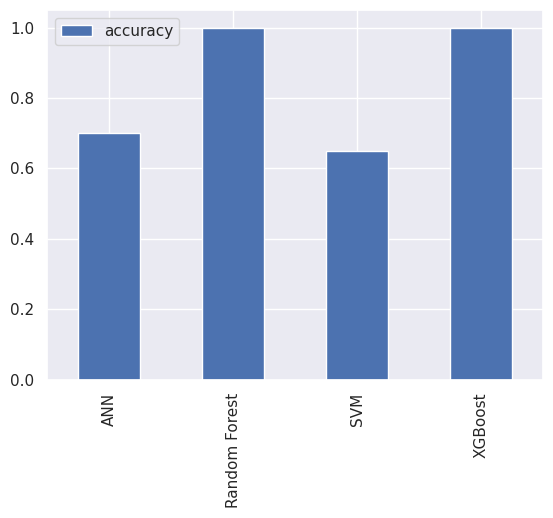
{'Sex': 0.165},

{'BP': 0.0},

{'Cholesterol': 0.434},

{'Na\_to\_K': 0.0}]

{'ANN': 0.7, 'Random Forest': 1.0, 'SVM': 0.65, 'XGBoost': 1.0}

****

Fitting 5 folds for each of 20 candidates, totalling 100 fits

* RandomizedSearchCV

RandomizedSearchCV(cv=5, estimator=RandomForestClassifier(), n\_iter=20,

param\_distributions={'max\_depth': [None, 3, 5, 10],

'min\_samples\_leaf': array([ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19]),

'min\_samples\_split': array([ 2, 4, 6, 8, 10, 12, 14, 16, 18]),

'n\_estimators': array([ 10, 60, 110, 160, 210, 260, 310, 360, 410, 460, 510, 560, 610,660, 710, 760, 810, 860, 910, 960])},

verbose=True)

estimator: RandomForestClassifier

RandomForestClassifier()

RandomForestClassifier

RandomForestClassifier()

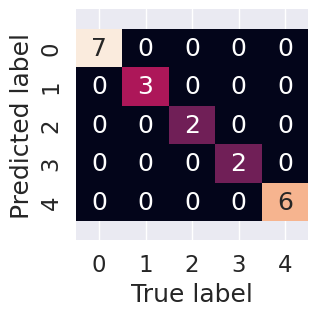
* {'n\_estimators': 510,

'min\_samples\_split': 14,

'min\_samples\_leaf': 1,

'max\_depth': None}

* 1.0



* precision recall f1-score support

0 1.00 1.00 1.00 3

1 1.00 1.00 1.00 2

2 1.00 1.00 1.00 2

3 1.00 1.00 1.00 6

4 1.00 1.00 1.00 7

* accuracy 1.00 20

macroavg 1.00 1.00 1.00 20

weightedavg 1.00 1.00 1.00 20