In [21]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.model_selection import train_test_split
```

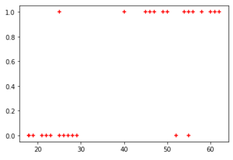
In [27]:
```
df = pd.read_csv('D:\\24 - Machine_Learning\\download files\\insurance_data.csv')
df
```

Out[27]:

	age	bought_insurance
0	22	0
1	25	0
2	47	1
3	52	0
4	46	1
5	56	1
6	55	0
7	60	1
8	62	1
9	61	1
10	18	0
11	28	0
12	27	0
13	29	0
14	49	1
15	55	1
16	25	1
17	58	1
18	19	0
19	18	0
20	21	0
21	26	0
22	40	1
23	45	1
24	50	1
25	54	1
26	23	0

In [4]:
```
plt.scatter(df.age, df.bought_insurance, marker = '+', color = 'red')
```

Out[4]:
```
<matplotlib.collections.PathCollection at 0x1f8346039a0>
```



In [5]:
```
X = df.drop('bought_insurance', axis='columns')
```

In [6]:
```
Y = df.drop('age', axis='columns')
```

In [12]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=41)
```

In [13]:
```
# LOGISTIC REGRESSION
LR = LogisticRegression()
LR.fit(X_train, Y_train)
```

```
D:\24-Annaconda\lib\site-packages\sklearn\utils\validation.py:993: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[13]:
```
LogisticRegression()
```

In [14]:
```
Y_pred = LR.predict(X_test)
Y_pred
```

Out[14]:
```
array([1, 0, 1, 1, 0, 0], dtype=int64)
```

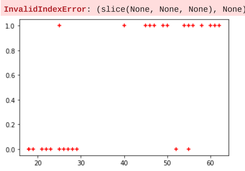
In [15]:
```
Y_test
```

Out[15]:

	bought_insurance
7	1
20	0
15	1
5	1
21	0
11	0

In [19]:
```
plt.scatter(df.age, df.bought_insurance, marker = '+', color = 'red')
plt.plot(X_test, Y_pred ,'b')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
File D:\24-Annaconda\lib\site-packages\pandas\core\indexes\base.py:3621, in Index.get_loc(self, key, method, tolerance)
   3620 try:
-> 3621     return self._engine.get_loc(casted_key)
   3622 except KeyError as err:

File D:\24-Annaconda\lib\site-packages\pandas\_libs\index.pyx:136, in pandas._libs.index.IndexEngine.get_loc()

File D:\24-Annaconda\lib\site-packages\pandas\_libs\index.pyx:142, in pandas._libs.index.IndexEngine.get_loc()

TypeError: '(slice(None, None, None), None)' is an invalid key

During handling of the above exception, another exception occurred:

InvalidIndexError                         Traceback (most recent call last)
Input In [19], in <cell line: 2>()
      1 plt.scatter(df.age, df.bought_insurance, marker = '+', color = 'red')
----> 2 plt.plot(X_test, Y_pred ,'b')

File D:\24-Annaconda\lib\site-packages\matplotlib\pyplot.py:2757, in plot(scalex, scaley, data, *args, **kwargs)
   2755 @_copy_docstring_and_deprecators(Axes.plot)
   2756 def plot(*args, scalex=True, scaley=True, data=None, **kwargs):
-> 2757     return gca().plot(
   2758         *args, scalex=scalex, scaley=scaley,
   2759         **({"data": data} if data is not None else {}), **kwargs)

File D:\24-Annaconda\lib\site-packages\matplotlib\axes\_axes.py:1632, in Axes.plot(self, scalex, scaley, data, *args, **kwargs)
   1390 """
   1391 Plot y versus x as lines and/or markers.
   1392
   (...)
   1629 (``'green'``) or hex strings (``'#008000'``).
   1630 """
   1631 kwargs = cbook.normalize_kwargs(kwargs, mlines.Line2D)
-> 1632 lines = [*self._get_lines(*args, data=data, **kwargs)]
   1633 for line in lines:
   1634     self.add_line(line)

File D:\24-Annaconda\lib\site-packages\matplotlib\axes\_base.py:312, in _process_plot_var_args.__call__(self, data, *args, **kwargs)
    310     this += args[0],
    311     args = args[1:]
--> 312 yield from self._plot_args(this, kwargs)

File D:\24-Annaconda\lib\site-packages\matplotlib\axes\_base.py:487, in _process_plot_var_args._plot_args(self, tup, kwargs, return_kwargs)
    484         kw[prop_name] = val
    486 if len(xy) == 2:
--> 487     x = _check_1d(xy[0])
    488     y = _check_1d(xy[1])
    489 else:

File D:\24-Annaconda\lib\site-packages\matplotlib\cbook\__init__.py:1327, in _check_1d(x)
   1321 with warnings.catch_warnings(record=True) as w:
   1322     warnings.filterwarnings(
   1323         "always",
   1324         category=Warning,
   1325         message='Support for multi-dimensional indexing')
-> 1327     ndim = x[:, None].ndim
   1328     # we have definitely hit a pandas index or series object
   1329     # cast to a numpy array.
   1330     if len(w) > 0:

File D:\24-Annaconda\lib\site-packages\pandas\core\frame.py:3505, in DataFrame.__getitem__(self, key)
   3503 if self.columns.nlevels > 1:
   3504     return self._getitem_multilevel(key)
-> 3505 indexer = self.columns.get_loc(key)
   3506 if is_integer(indexer):
   3507     indexer = [indexer]

File D:\24-Annaconda\lib\site-packages\pandas\core\indexes\base.py:3628, in Index.get_loc(self, key, method, tolerance)
   3623         raise KeyError(key) from err
   3624     except TypeError:
   3625         # If we have a listlike key, _check_indexing_error will raise
   3626         #  InvalidIndexError. Otherwise we fall through and re-raise
   3627         #  the TypeError.
-> 3628         self._check_indexing_error(key)
   3629         raise
   3631 # GH#42269

File D:\24-Annaconda\lib\site-packages\pandas\core\indexes\base.py:5637, in Index._check_indexing_error(self, key)
   5633 def _check_indexing_error(self, key):
   5634     if not is_scalar(key):
   5635         # if key is not a scalar, directly raise an error (the code below
   5636         # would convert to numpy arrays and raise later any way) - GH29926
-> 5637         raise InvalidIndexError(key)

InvalidIndexError: (slice(None, None, None), None)
```



In [17]:
```
LR.score(X_test, Y_test)
```

Out[17]:
```
1.0
```

In [20]:
```
pred = np.array(Y_pred)
xtest = np.array(X_test)
xtest = xtest.reshape(1,-1)
xtest.shape
```

Out[20]:
```
(1, 6)
```

In [22]:
```
# CLASSIFICATION REPORT
print(classification_report(Y_test, Y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         3
           1       1.00      1.00      1.00         3

    accuracy                           1.00         6
   macro avg       1.00      1.00      1.00         6
weighted avg       1.00      1.00      1.00         6
```

In [23]:
```
# CONFUSION MATR
print(confusion_matrix(Y_test, Y_pred))
```

```
[[3 0]
 [0 3]]
```