In [2]:
```python
import pandas as pd
```

In [3]:
```python
df=pd.read_csv("C:/Users/dilee/Desktop/SSV/Vijay/career_success.csv")
```

In [4]:
```python
df
```

Out[4]:

	age	current_job	skillset	interests	responsibilities	qualification	career_success	class
0	20	4	2	5	3	5	6.6	1
1	20	1	2	5	2	3	5.9	0
2	22	2	2	7	1	4	6.5	1
3	25	3	1	8	2	3	7.1	1
4	20	2	1	4	2	1	5.4	0
...	...	...	...	...	...	...	...	...
9995	20	2	2	9	1	1	6.0	0
9996	21	5	5	4	3	2	7.4	1
9997	20	4	2	3	3	5	6.4	1
9998	24	5	5	3	3	5	8.2	1
9999	23	1	2	9	3	1	6.9	1

10000 rows × 8 columns

In [5]:
```python
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
```

In [21]:
```python
X = df[['age','current_job','skillset','interests','responsibilities']]
Y = df['career_success']
Y_=df['class']
```

In [48]:
```python
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error,mean_squared_error,r2_score

#LINEAR REGRESSION
mdl = LinearRegression()
x_train,x_test,y_train,y_test=train_test_split(X,Y,test_size=0.2,random_state=1)
mdl.fit(x_train, y_train)
pred = mdl.predict(x_test)
```

In [23]:
```python
X
```

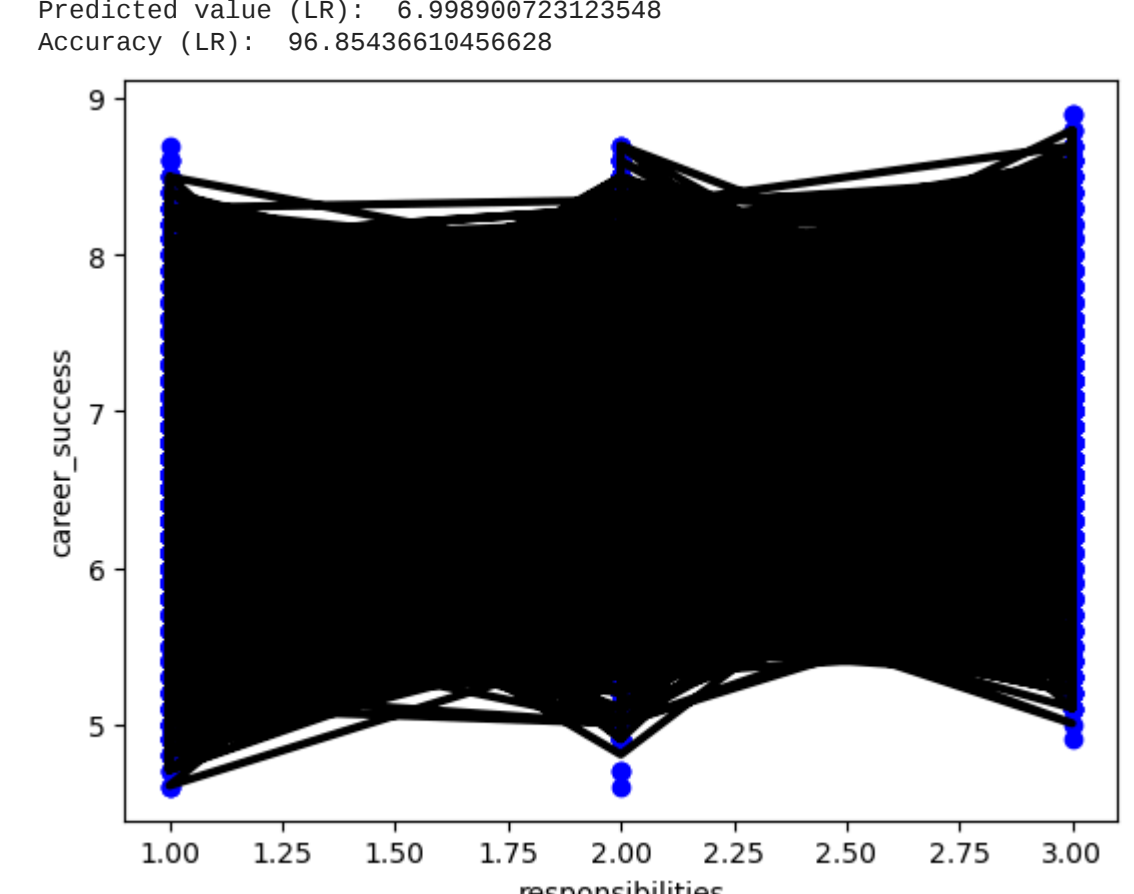
Out[23]:

	age	current_job	skillset	interests	responsibilities
0	20	4	2	5	3
1	20	1	2	5	2
2	22	2	2	7	1
3	25	3	1	8	2
4	20	2	1	4	2
...	...	...	...	...	...
9995	20	2	2	9	1
9996	21	5	5	4	3
9997	20	4	2	3	3
9998	24	5	5	3	3
9999	23	1	2	9	3

10000 rows × 5 columns

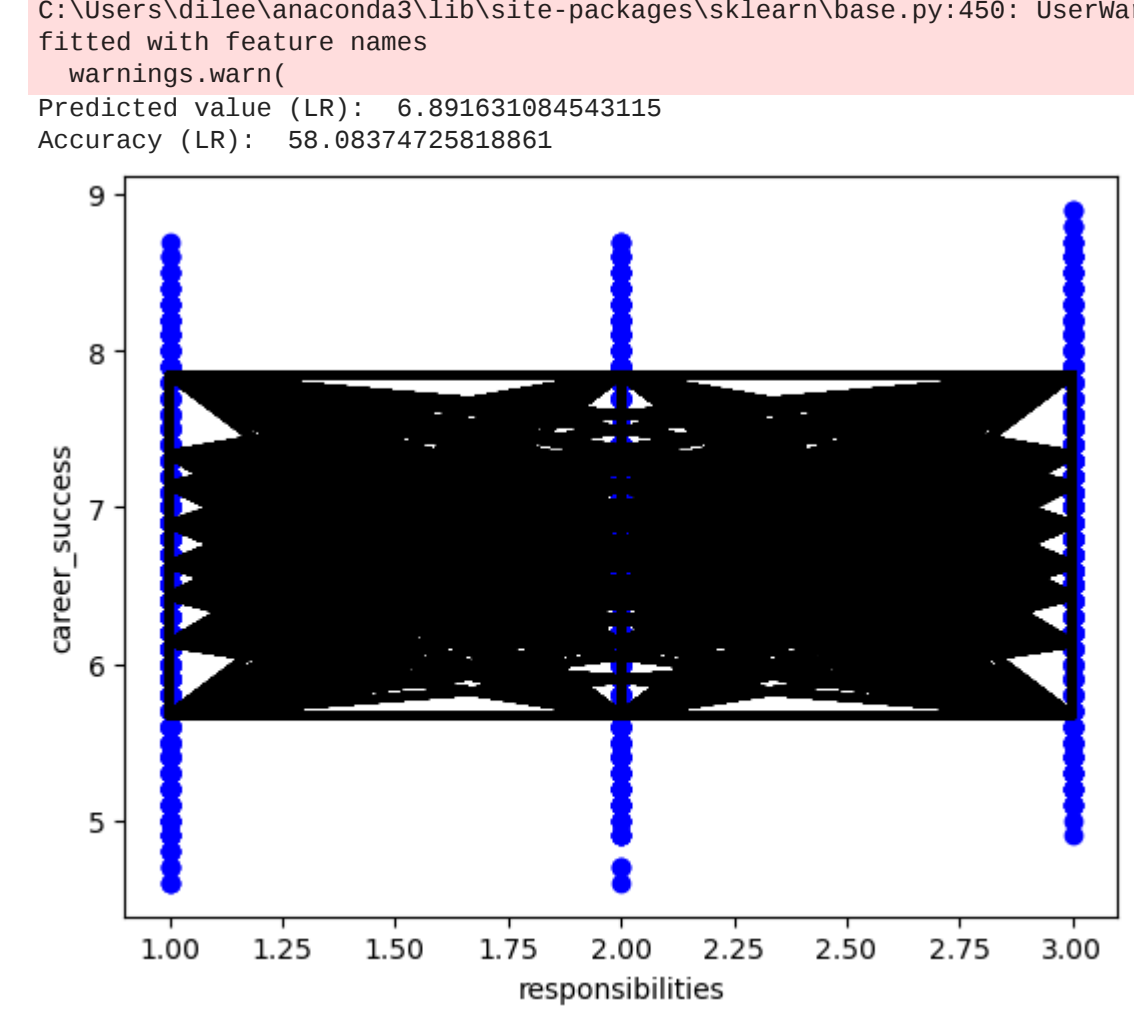
In [54]:
```python
mdl = LinearRegression()
mdl.fit(x_train, y_train)
pred = mdl.predict([[21,3,3,7,3]])
print("Predicted value (LR): ",pred[0])
print("Accuracy (LR): ",mdl.score(X[:100], Y[:100])*100)

plt.scatter(X['responsibilities'], Y, color='b')
plt.plot(X['responsibilities'], mdl.predict(X),color='black',linewidth=3)
plt.xlabel('responsibilities')
plt.ylabel('career_success')
plt.show()
```

```
C:\Users\dilee\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitt
ed with feature names
  warnings.warn(
Predicted value (LR):  6.998900723123548
Accuracy (LR):  96.85436610456628
```



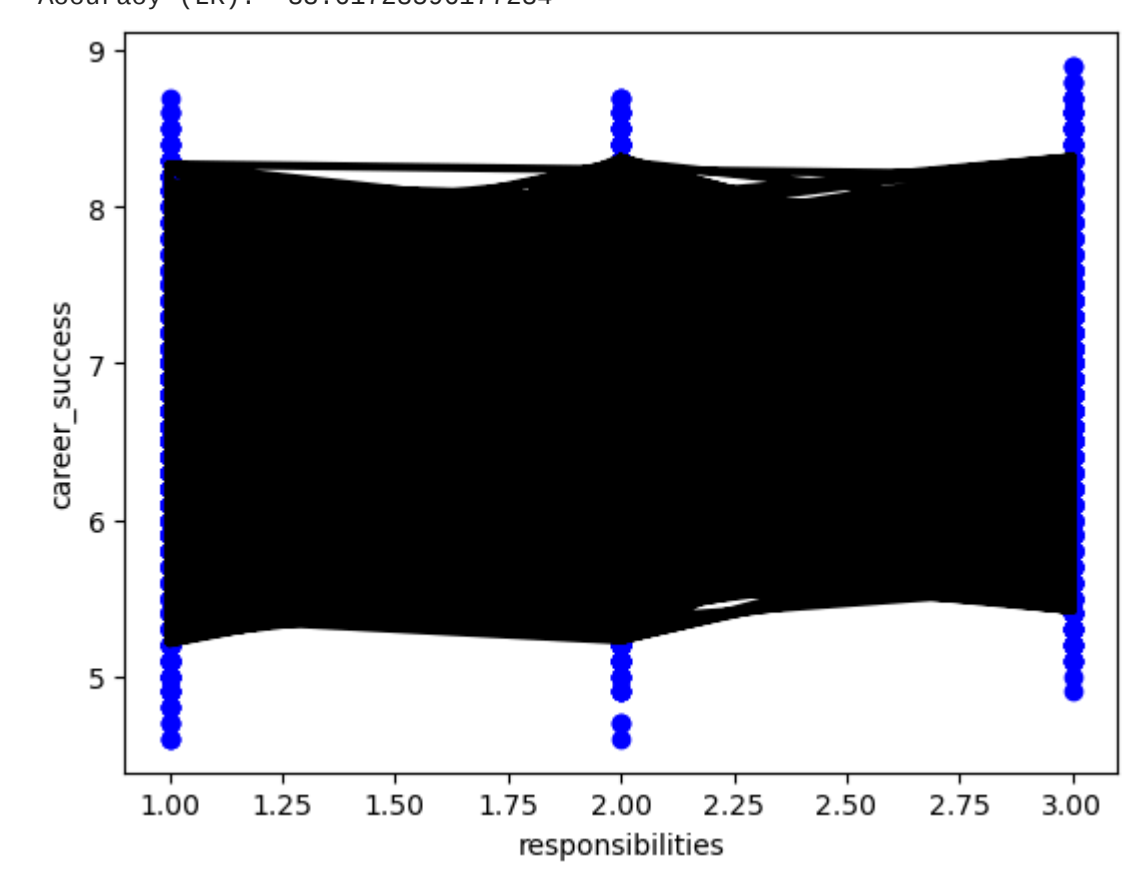
In [29]:
```python
from sklearn.tree import DecisionTreeRegressor
mdl =  DecisionTreeRegressor(max_depth=3)
mdl.fit(x_train, y_train)
pred = mdl.predict([[21,3,3,7,3]])
print("Predicted value (LR): ",pred[0])
print("Accuracy (LR): ",mdl.score(X[:100], Y[:100])*100)

plt.scatter(X['responsibilities'], Y, color='b')
plt.plot(X['responsibilities'], mdl.predict(X),color='black',linewidth=3)
plt.xlabel('responsibilities')
plt.ylabel('career_success')
plt.show()
```

```
C:\Users\dilee\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but DecisionTreeRegressor was
fitted with feature names
  warnings.warn(
Predicted value (LR):  6.891631084543115
Accuracy (LR):  58.08374725818861
```



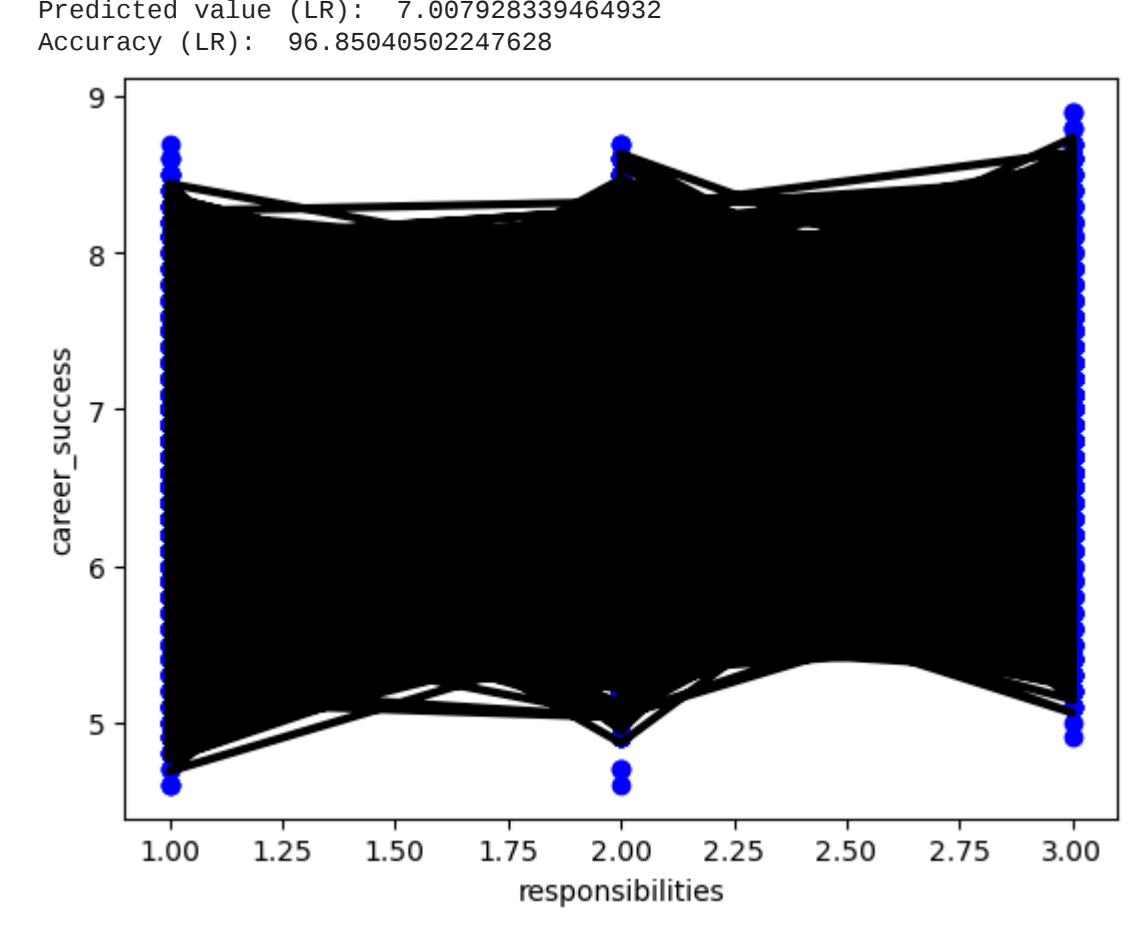
In [31]:
```python
from sklearn.ensemble import RandomForestRegressor
mdl = RandomForestRegressor(n_estimators=100,max_depth=6)
mdl.fit(x_train, y_train)
pred = mdl.predict([[21,3,3,7,3]])
print("Predicted value (LR): ",pred[0])
print("Accuracy (LR): ",mdl.score(X[:100], Y[:100])*100)

plt.scatter(X['responsibilities'], Y, color='b')
plt.plot(X['responsibilities'], mdl.predict(X),color='black',linewidth=3)
plt.xlabel('responsibilities')
plt.ylabel('career_success')
plt.show()
```

```
C:\Users\dilee\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but RandomForestRegressor was
fitted with feature names
  warnings.warn(
Predicted value (LR):  6.929404613629257
Accuracy (LR):  88.61728596177284
```



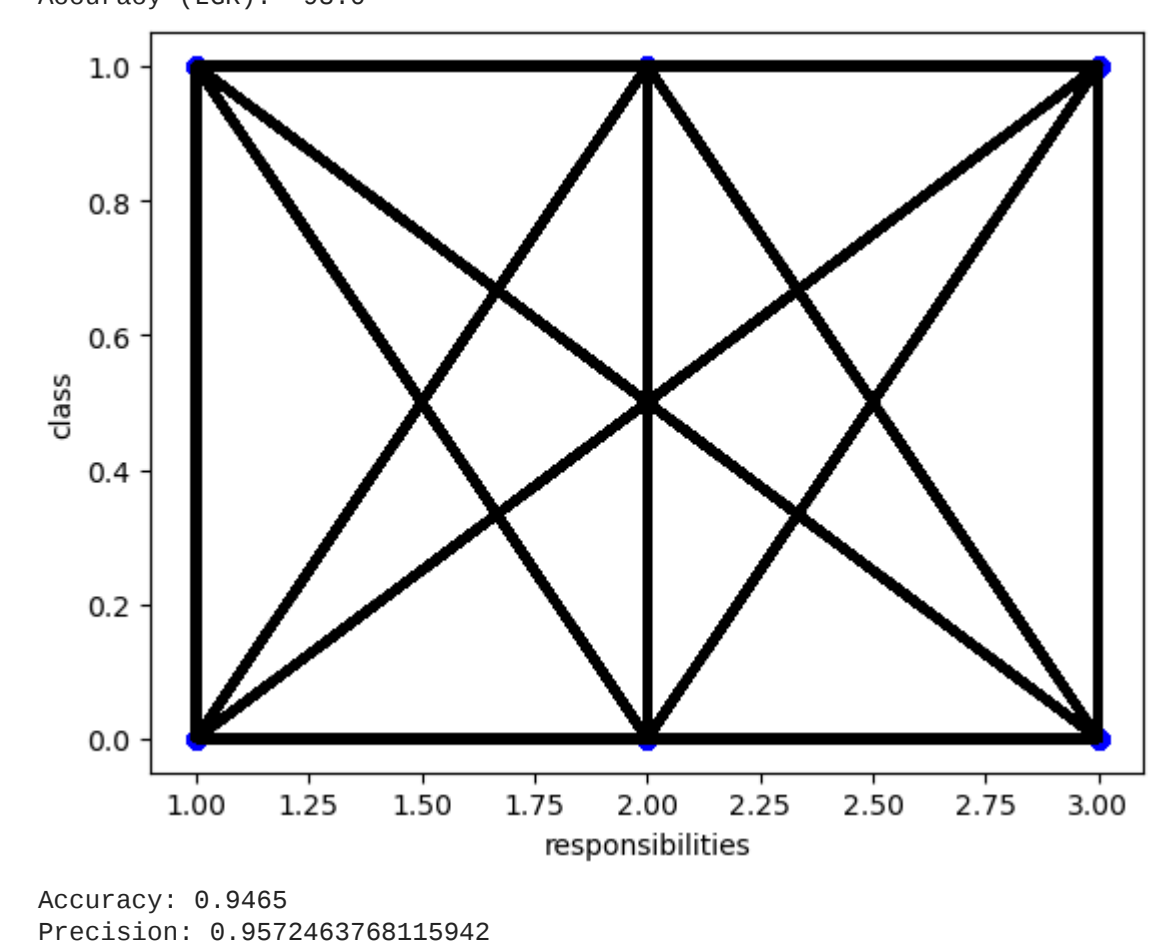
In [39]:
```python
from sklearn.svm import SVR
mdl = SVR(kernel = 'rbf')
mdl.fit(x_train, y_train)
pred = mdl.predict([[21,3,3,7,3]])
print("Predicted value (LR): ",pred[0])
print("Accuracy (LR): ",mdl.score(X[:100], Y[:100])*100)

plt.scatter(X['responsibilities'], Y, color='b')
plt.plot(X['responsibilities'], mdl.predict(X),color='black',linewidth=3)
plt.xlabel('responsibilities')
plt.ylabel('career_success')
plt.show()
```

```
C:\Users\dilee\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but SVR was fitted with featu
re names
  warnings.warn(
Predicted value (LR):  7.007928339464932
Accuracy (LR):  96.85040502247628
```



In [58]:
```python
#Evaluation Metrics for classification
from sklearn.linear_model import LogisticRegression
x_train,x_test,y_train,y_test=train_test_split(X,Y_,test_size=0.2,random_state=1)

mdl = LogisticRegression()
mdl.fit(x_train, y_train)

pred = mdl.predict(x_test)
print("Predicted value (LGR): ",pred[1])
print("Accuracy (LGR): ",mdl.score(X[:100], Y_[:100])*100)

plt.scatter(X['responsibilities'], Y_, color='b')
plt.plot(X['responsibilities'], mdl.predict(X),color='black',linewidth=3)
plt.xlabel('responsibilities')
plt.ylabel('class')
plt.show()

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# Accuracy
accuracy = accuracy_score(y_test, pred)
print("Accuracy:", accuracy)

# Precision
precision = precision_score(y_test, pred)
print("Precision:", precision)

# Recall
recall = recall_score(y_test, pred)
print("Recall:", recall)

# F1 Score
f1 = f1_score(y_test, pred)
print("F1 Score:", f1)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, pred)
print("Confusion Matrix:")
print(conf_matrix)
```

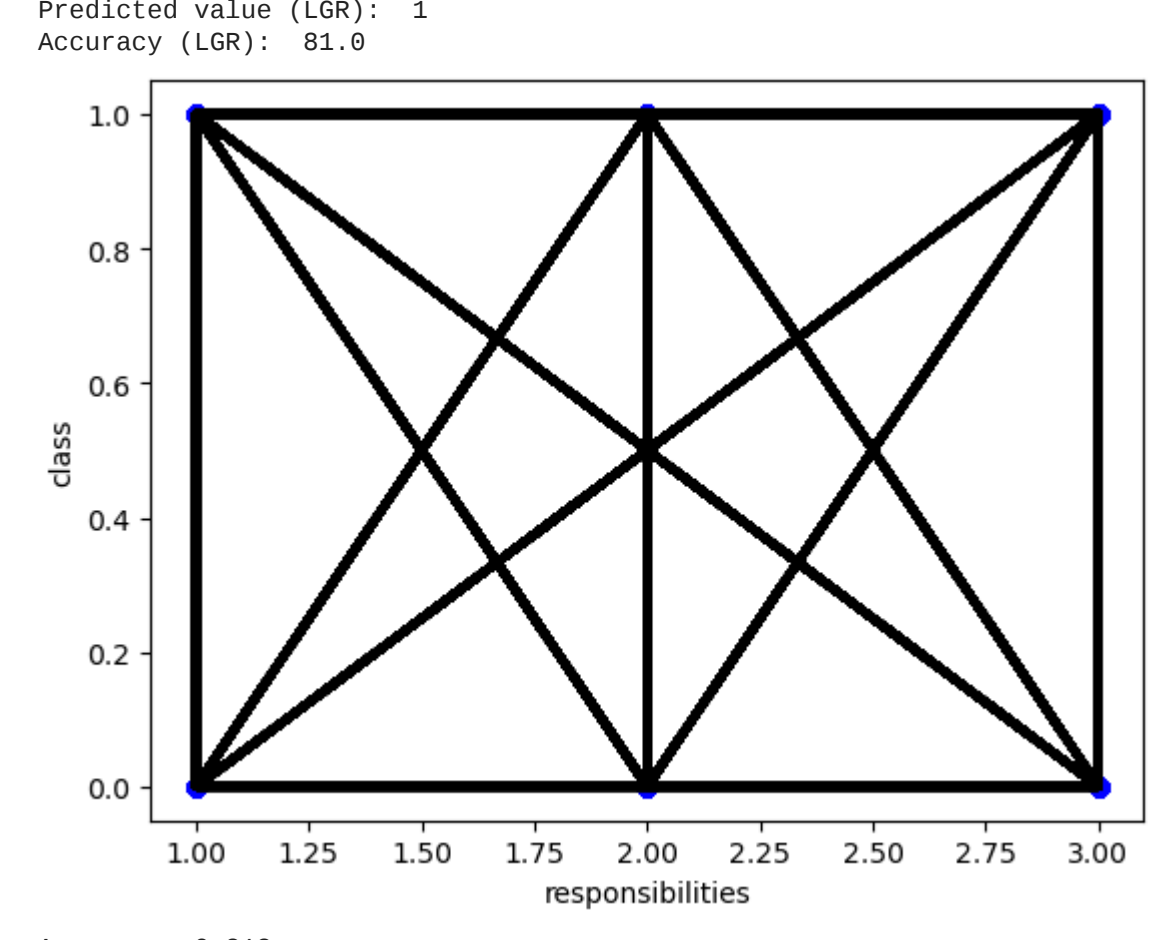
```
Predicted value (LGR):  1
Accuracy (LGR):  98.0
```




```
Accuracy: 0.9465
Precision: 0.9572463768115942
Recall: 0.9649379108838568
F1 Score: 0.9610767551837032
Confusion Matrix:
[[ 572   59]
 [  48 1321]]
```

In [59]:
```python
#DECISION TREE CLASSIFICATION
from sklearn.tree import DecisionTreeClassifier
mdl = DecisionTreeClassifier(max_leaf_nodes=3, random_state=1)
mdl.fit(x_train, y_train)

pred = mdl.predict(x_test)
print("Predicted value (LGR): ",pred[0])
print("Accuracy (LGR): ",mdl.score(X[:100], Y_[:100])*100)

plt.scatter(X['responsibilities'], Y_, color='b')
plt.plot(X['responsibilities'], mdl.predict(X),color='black',linewidth=3)
plt.xlabel('responsibilities')
plt.ylabel('class')
plt.show()
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix

# Accuracy
accuracy = accuracy_score(y_test, pred)
print("Accuracy:", accuracy)

# Precision
precision = precision_score(y_test, pred)
print("Precision:", precision)

# Recall
recall = recall_score(y_test, pred)
print("Recall:", recall)

# F1 Score
f1 = f1_score(y_test, pred)
print("F1 Score:", f1)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test, pred)
print("Confusion Matrix:")
print(conf_matrix)
```

```
Predicted value (LGR):  1
Accuracy (LGR):  81.0
```




```
Accuracy: 0.818
Precision: 0.8214971209213052
Recall: 0.9379108838568299
F1 Score: 0.8758526603001365
Confusion Matrix:
[[ 352  279]
 [  85 1284]]
```

In [51]:
```python
#Evaluation Metrics for Regration
from sklearn.metrics import mean_absolute_error
from sklearn.linear_model import LinearRegression
regressor=LinearRegression()
regressor.fit(x_train,y_train)
score=regressor.score(x_train,y_train)
y_pred=regressor.predict(x_test)
y_pred=regressor.predict(x_test)

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
mse=mean_squared_error(y_test,y_pred)
r2=r2_score(y_test,y_pred)
mae=mean_absolute_error(y_test,y_pred)
print("Mean Square Error of Linear  Regression",mse)
print("R2 score of Linear Regression",r2)
print("Mean absolute Error of Linear Regression",mae)
```

```
Mean Square Error of Linear  Regression 0.02002712445270837
R2 score of Linear Regression 0.9639591868487457
Mean absolute Error of Linear Regression 0.12093222855285952
```

In [52]:
```python
from sklearn.ensemble import RandomForestRegressor
rf_model=RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(x_train, y_train)
y_pred=rf_model.predict(x_test)

from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
mse=mean_squared_error(y_test,y_pred)
r2=r2_score(y_test,y_pred)
mae=mean_absolute_error(y_test,y_pred)
print("Mean Square Error of RandomForest Regression",mse)
print("R2 score of RandomForest Regression",r2)
print("Mean absolute Error of RandomForest Regression",mae)
```

```
Mean Square Error of RandomForest Regression 0.027046914311033367
R2 score of RandomForest Regression 0.951326373024555
Mean absolute Error of RandomForest Regression 0.1372503181074643
```

In [55]:
```python
from sklearn.svm import SVR
svr=SVR(kernel='rbf')
svr.fit(x_train,y_train)

# Make predictions on the test set
y_preds=svr.predict(x_test)

# Evaluate the model
mse=mean_squared_error(y_test,y_preds)
r2=r2_score(y_test,y_pred)
mae=mean_absolute_error(y_test,y_preds)
print("Mean Square Error of Support Vector Regression",mse)
print("R2 score of Support Vector Regression",r2)
print("Mean absolute Error of Support Vector Regression",mae)
```

```
Mean Square Error of Support Vector Regression 0.020182708559763923
R2 score of Support Vector Regression 0.951326373024555
Mean absolute Error of Support Vector Regression 0.12178191517013928
```

In [57]:
```python
from sklearn.tree import DecisionTreeRegressor
mdl =  DecisionTreeRegressor(max_depth=3)
mdl.fit(x_train,y_train)
y_predr=mdl.predict(x_test)

# Evaluate the model
mse=mean_squared_error(y_test,y_predr)
r2=r2_score(y_test,y_predr)
mae=mean_absolute_error(y_test,y_predr)
print("Mean Square Error of Support Vector Regression",mse)
print("R2 score of Support Vector Regression",r2)
print("Mean absolute Error of Support Vector Regression",mae)
```

```
Mean Square Error of Support Vector Regression 0.18602079974884514
R2 score of Support Vector Regression 0.6652369689005273
Mean absolute Error of Support Vector Regression 0.3445154520727665
```

In [ ]: