In [64]:

```
import pandas as pd
```

In [65]:

```
df = pd. read_csv('exp4.csv')
df.head()
```

Out[65]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	(
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	

In [66]:

```
df.drop(['PassengerId', 'Name', 'SibSp', 'Parch', 'Ticket', 'Cabin', 'Embarked'], axis='col
df.head()
```

Out[66]:

	Survived	Pclass	Sex	Age	Fare
0	0	3	male	22.0	7.2500
1	1	1	female	38.0	71.2833
2	1	3	female	26.0	7.9250
3	1	1	female	35.0	53.1000
4	0	3	male	35.0	8.0500

In [67]:

```
inputs = df.drop('Survived', axis='columns')
inputs.head()
```

Out[67]:

	Pclass	Sex	Age	Fare
0	3	male	22.0	7.2500
1	1	female	38.0	71.2833
2	3	female	26.0	7.9250
3	1	female	35.0	53.1000
4	3	male	35.0	8.0500

In [68]:

```
final_inputs = pd.get_dummies(inputs, columns=['Sex'])
final_inputs.head()
```

Out[68]:

	Pclass	Age	Fare	Sex_female	Sex_male
0	3	22.0	7.2500	0	1
1	1	38.0	71.2833	1	0
2	3	26.0	7.9250	1	0
3	1	35.0	53.1000	1	0
4	3	35.0	8.0500	0	1

In [69]:

```
final_inputs.Age = final_inputs.Age.fillna(inputs.Age.mean())
final_inputs
```

Out[69]:

	Pclass	Age	Fare	Sex_female	Sex_male
0	3	22.000000	7.2500	0	1
1	1	38.000000	71.2833	1	0
2	3	26.000000	7.9250	1	0
3	1	35.000000	53.1000	1	0
4	3	35.000000	8.0500	0	1
...	...	...	...	...	...
886	2	27.000000	13.0000	0	1
887	1	19.000000	30.0000	1	0
888	3	29.699118	23.4500	1	0
889	1	26.000000	30.0000	0	1
890	3	32.000000	7.7500	0	1

891 rows × 5 columns

In [70]:

```
target = df.Survived
target.head()
```

Out[70]:

```
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

In [71]:

```
from sklearn import preprocessing
```

In [72]:

```
d = preprocessing.normalize(final_inputs)
df_final = pd.DataFrame(d, columns=["Pclass","Age","Fare","Sex_female","Sex_male"])
df_final.head()
```

Out[72]:

	Pclass	Age	Fare	Sex_female	Sex_male
0	0.128322	0.941028	0.310112	0.000000	0.042774
1	0.012377	0.470345	0.882309	0.012377	0.000000
2	0.109632	0.950143	0.289611	0.036544	0.000000
3	0.015720	0.550202	0.834735	0.015720	0.000000
4	0.083211	0.970799	0.223284	0.000000	0.027737

In [73]:

```
from sklearn.model_selection import train_test_split
```

In [74]:

```
x_train, x_test, y_train, y_test = train_test_split(final_inputs, target, test_size=0.3)
```

In [75]:

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from vecstack import stacking
from sklearn.metrics import mean_squared_error, accuracy_score
from math import sqrt
```

In [76]:

```
# Stacking

# import numpy as np
# y_train = np.array(y_train)

model1 = LogisticRegression()
model2 = KNeighborsClassifier()
model3 = DecisionTreeClassifier()

all_models = [model1, model2, model3]

s_train, s_test = stacking(all_models, x_train, y_train, x_test, regression=True, random_st

final_model = model1

final_model.fit(s_train, y_train)
pred = final_model.predict(s_test)

print("Root mean square error = ", sqrt(mean_squared_error(y_test, pred)))
print("Accuracy = ", accuracy_score(y_test, pred))
```

```
Root mean square error =  0.41877575661487165
Accuracy =  0.8246268656716418
```

In [77]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [78]:

```
# Bagging
clf = RandomForestClassifier()

clf.fit(x_train, y_train)
pred = clf.predict(x_test)

print("Root mean square error = ", sqrt(mean_squared_error(y_test, pred)))
print("Accuracy = ", accuracy_score(y_test, pred))
```

```
Root mean square error =  0.41877575661487165
Accuracy =  0.8246268656716418
```

In [79]:

```
from xgboost import XGBClassifier
```

In [80]:

```
# Boosting
clf = XGBClassifier()

clf.fit(x_train, y_train)
pred = clf.predict(x_test)

print("Root mean square error = ", sqrt(mean_squared_error(y_test, pred)))
print("Accuracy = ", accuracy_score(y_test, pred))
```

```
Root mean square error =  0.42759305524706830
Accuracy =  0.8171641791044776
```