**SVM**

**Import packages:**

import pandas as pd

from sklearn import preprocessing

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import accuracy\_score

from sklearn.metrics import confusion\_matrix

from sklearn import svm

import numpy as np

#**Import data:**

dataset=pd.read\_csv('train.csv')

dataset.columns

del dataset["Ticket"]

del dataset["Cabin"]

del dataset["Name"]

del dataset["PassengerId"]

**Convert catagorical to numeric:**

le=preprocessing.LabelEncoder()

le.fit(dataset["Sex"])

dataset["Sex"]=le.transform(dataset["Sex"])

dataset["Fare"]=np.round(dataset["Fare"])

dataset["Age"]=np.round(dataset["Age"])

dataset["Embarked"]=le.fit\_transform(dataset["Embarked"])

**Split data x and y:**

y=dataset["Survived"]

x=dataset.drop(["Survived"],axis=1)

**Split the data:**

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.3,random\_state=0)

**Fit the model:**

clf=svm.SVC(gamma=0.01,C=100)

clf.fit(xtrain,ytrain)

ypre=clf.predict(xtest)

print(accuracy\_score(ytest, ypre,normalize=True))

print(confusion\_matrix(ytest, ypre))

0.7303370786516854

[[123 34]

[ 38 72]]

**Using for loop for find higher accuracy:**

def accuracy\_var(y,x):

xtrain,xtest,ytrain,ytest=train\_test\_split(x,y,test\_size=0.3,random\_state=0)

clf=svm.SVC(gamma=0.01,C=100)

y\_pre=clf.fit(xtrain,ytrain).predict(xtest)

print(accuracy\_score(ytest, ypre,normalize=True))

print(confusion\_matrix(ytest, ypre))

k=list(["Pclass","Survived","Sex","SibSp","Parch","Embarked"])

for i in k:

y=pd.DataFrame(dataset[i]);

x=dataset.drop([i],axis=1);

accuracy\_var(y,x)

0.8876404494382022

[[ 64 4 2]

[ 4 34 11]

[ 1 8 139]]

0.16853932584269662

[[ 0 24 24 109]

[ 0 45 22 43]

[ 0 0 0 0]

[ 0 0 0 0]]

0.14606741573033707

[[ 0 30 17 51]

[ 0 39 29 101]

[ 0 0 0 0]

[ 0 0 0 0]]

0.12734082397003746

[[ 0 39 36 107 0 0 0]

[ 0 27 9 29 0 0 0]

[ 0 2 1 5 0 0 0]

[ 0 1 0 6 0 0 0]

[ 0 0 0 2 0 0 0]

[ 0 0 0 1 0 0 0]

[ 0 0 0 2 0 0 0]]

0.07116104868913857

[[ 0 42 35 121 0]

[ 0 15 8 17 0]

[ 0 12 3 12 0]

[ 0 0 0 1 0]

[ 0 0 0 1 0]]

0.15730337078651685

[[ 0 28 4 17]

[ 0 0 0 22]

[ 0 41 42 113]

[ 0 0 0 0]]

#here pclass have higher accuracy