

**NAME**: Syed Naqi Haider

**ID**  : F2017065126

**COURSE:** DATA SCIENCE TECHNOLOGIES

**SUBJECT**: PROJECT DOCUMENTATION

**TO**  : MAZHAR JAVED AWAN

**PROJECT DOCUMENTATION**

The Data set which used is Sindh-School-Enrollment-Stats after seen it with quite detail and try to find the best usefull data from It I have decided to use it to predicte the number of teacher required for a gaven amount of students and institution. I have tried to explain the working and did my best to perform all the task.

import pandas as pd

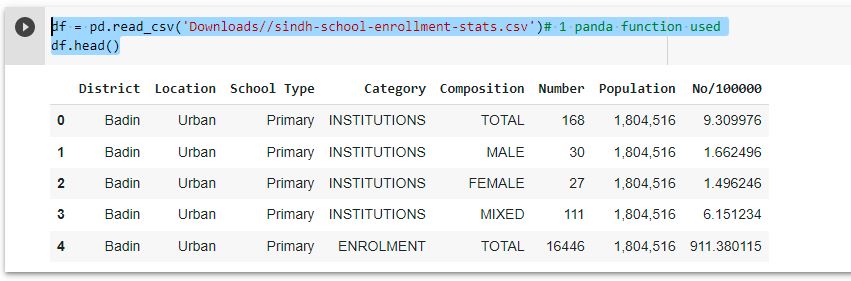
import matplotlib.pyplot as plt

import seaborn as sns

import numpy as np # all the required libaries are Imported for working

df = pd.read\_csv('Downloads//sindh-school-enrollment-stats.csv')# 1 panda function used

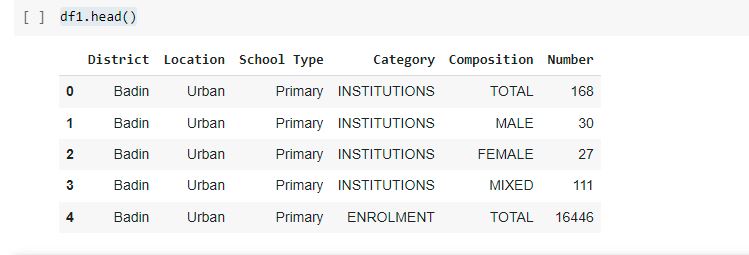
df.head()



Now We start cleaning and removing all the data which is not requried for prediction .

df1 = df.drop(['Population', 'No/100000'], axis=1)#2 function is used of  panda

df1.head()



A\_filter = (df1['Category'] == 'INSTITUTIONS') & (df1['Composition'] == 'TOTAL')

df2 = df1[A\_filter]

s=np. arange(348)#1 numpy function used

df2.set\_index(s, inplace=True)

A\_filter = (df1['Category'] == 'TEACHING STAFF') & (df1['Composition'] == 'TOTAL')

df3 = df1[A\_filter]

df3.set\_index(s, inplace=True)

A\_filter = (df1['Category'] == 'ENROLMENT') & (df1['Composition'] == 'TOTAL')

df4 = df1[A\_filter]

df4.set\_index(s, inplace=True)

df5 = df1.drop(['Category', 'Composition', 'Number','School Type'], axis=1)

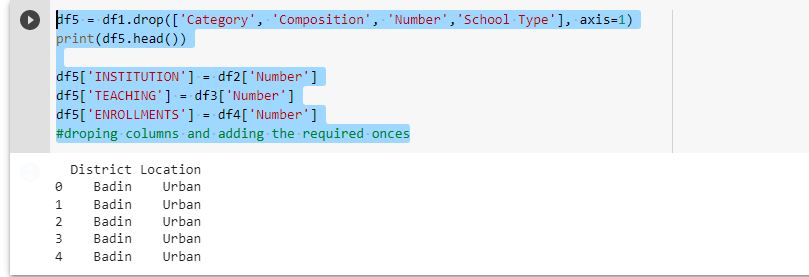
print(df5.head())

df5['INSTITUTION'] = df2['Number']

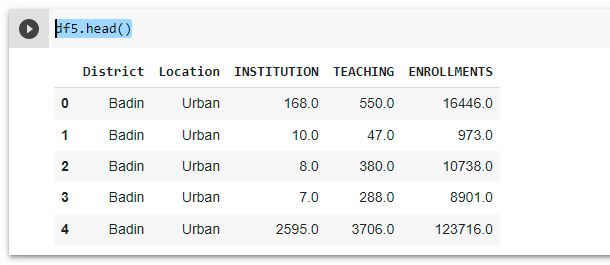
df5['TEACHING'] = df3['Number']

df5['ENROLLMENTS'] = df4['Number']

#droping columns and adding the required onces



df5.head()



df6=df5.dropna(how='any')# final datafram which is going to be used to apply the graphics and models

df6.head()

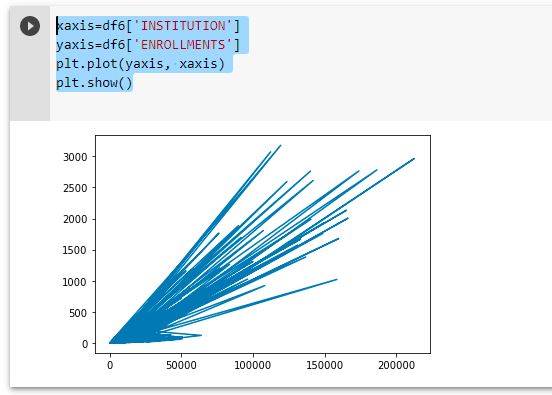
Matplotlib

xaxis=df6['INSTITUTION']

yaxis=df6['ENROLLMENTS']

plt.plot(yaxis, xaxis)

plt.show()

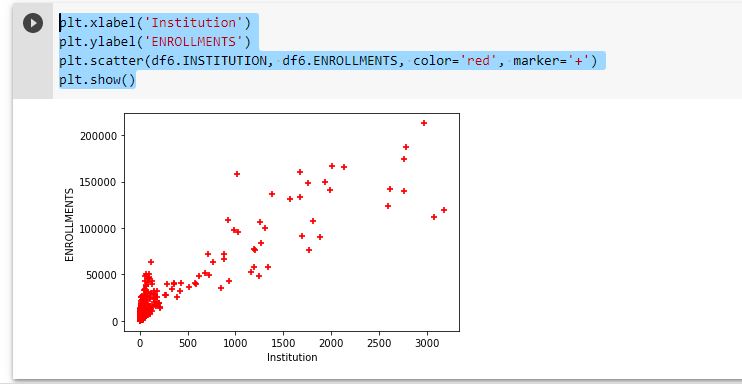


plt.xlabel('Institution')

plt.ylabel('ENROLLMENTS')

plt.scatter(df6.INSTITUTION, df6.ENROLLMENTS, color='red', marker='+')

plt.show()



y = df6['TEACHING'].sum()

x= df6['ENROLLMENTS'].sum()

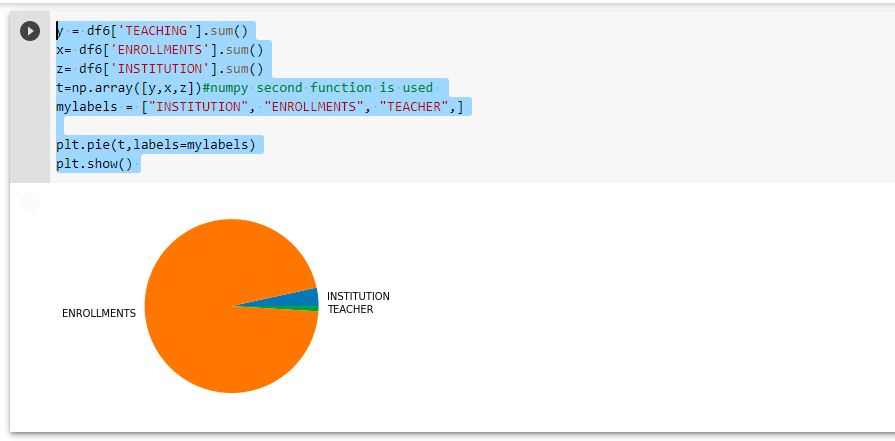
z= df6['INSTITUTION'].sum()

t=np.array([y,x,z])#numpy second function is used

mylabels = ["INSTITUTION", "ENROLLMENTS", "TEACHER",]

plt.pie(t,labels=mylabels)

plt.show()



y = df6['TEACHING'].sum()

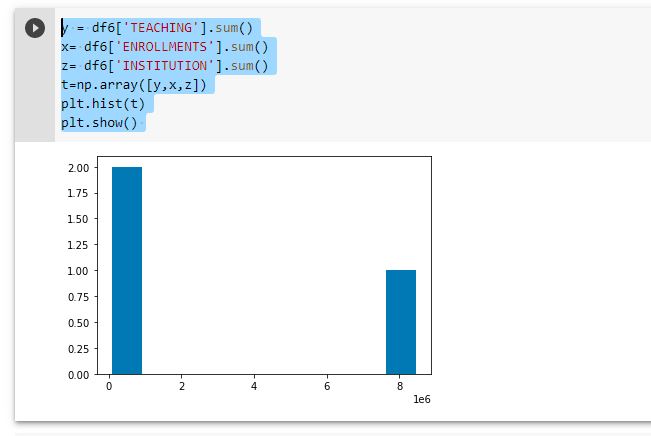
x= df6['ENROLLMENTS'].sum()

z= df6['INSTITUTION'].sum()

t=np.array([y,x,z])

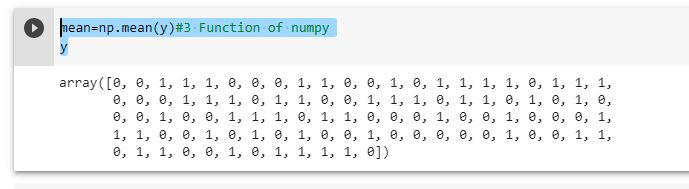
plt.hist(t)

plt.show()



mean=np.mean(y)#3 Function of numpy

y



y = df6['TEACHING'].sum()

x= df6['ENROLLMENTS'].sum()

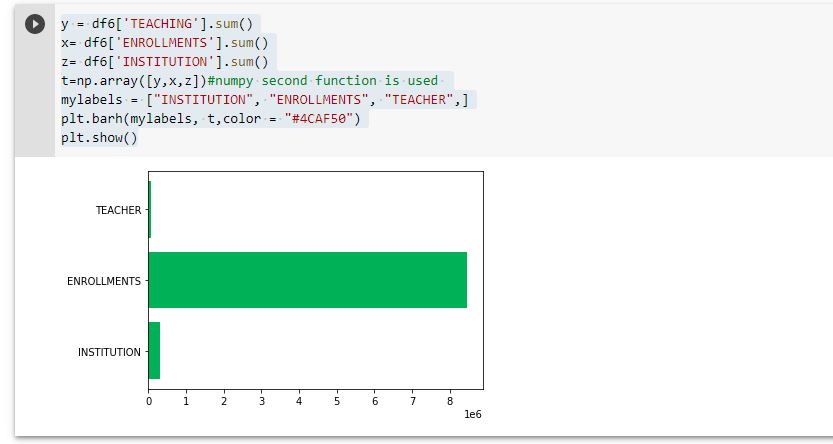
z= df6['INSTITUTION'].sum()

t=np.array([y,x,z])#numpy second function is used

mylabels = ["INSTITUTION", "ENROLLMENTS", "TEACHER",]

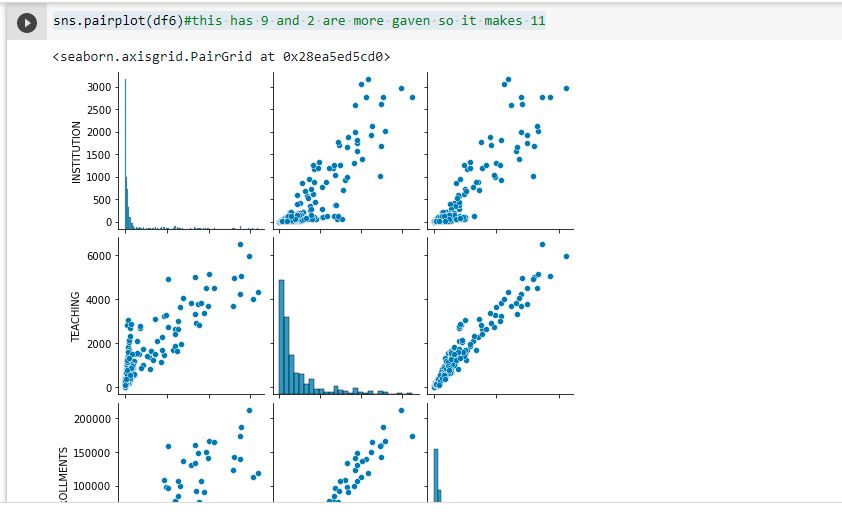
plt.barh(mylabels, t,color = "#4CAF50")

plt.show()

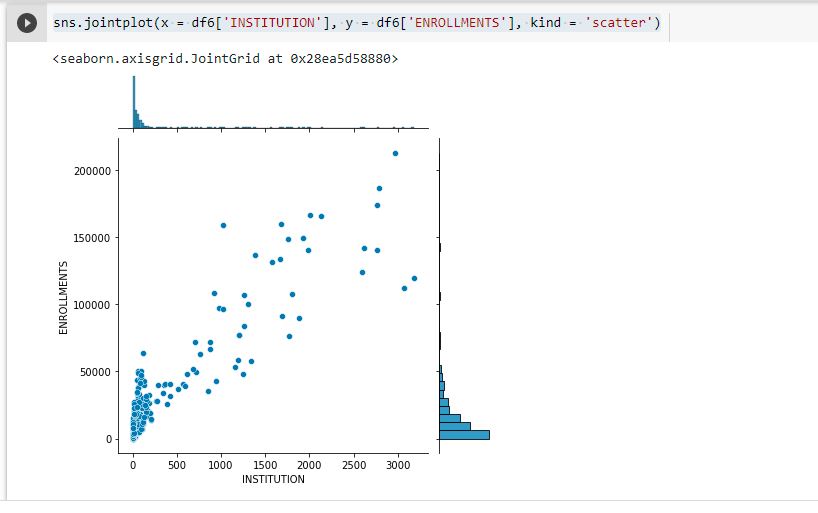


# SEABORN

sns.pairplot(df6)#this has 9 and 2 are more gaven so it makes 11



sns.jointplot(x = df6['INSTITUTION'], y = df6['ENROLLMENTS'], kind = 'scatter')



sns.distplot(x = df6['TEACHING'], bins = 10)



from sklearn import linear\_model

lr= linear\_model.LinearRegression()

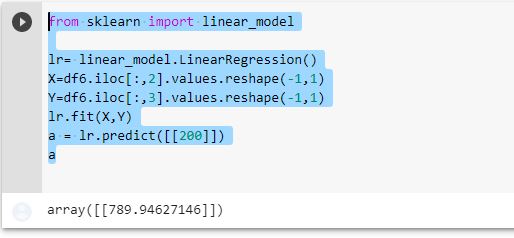
X=df6.iloc[:,2].values.reshape(-1,1)

Y=df6.iloc[:,3].values.reshape(-1,1)

lr.fit(X,Y)

a = lr.predict([[200]])

a



# Multi Regression

X=df6.drop(['District','Location','TEACHING'],axis=1)

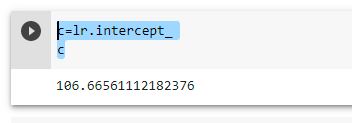
Y=df6['TEACHING']

lr.fit(X,Y)



c=lr.intercept\_

c



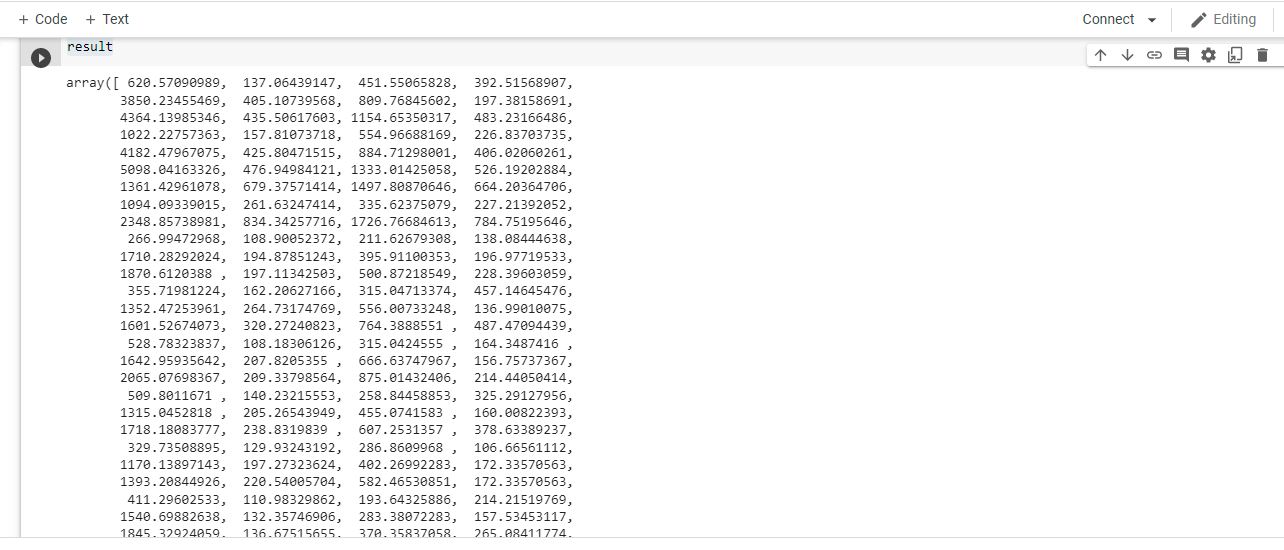
m=lr.coef\_

m



result=lr.predict(X)

result

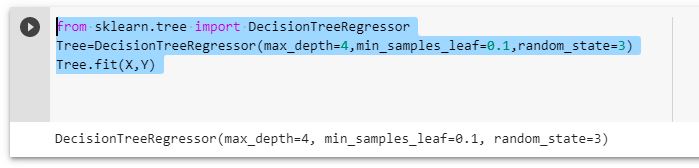


# Decision-Tree Regression

from sklearn.tree import DecisionTreeRegressor

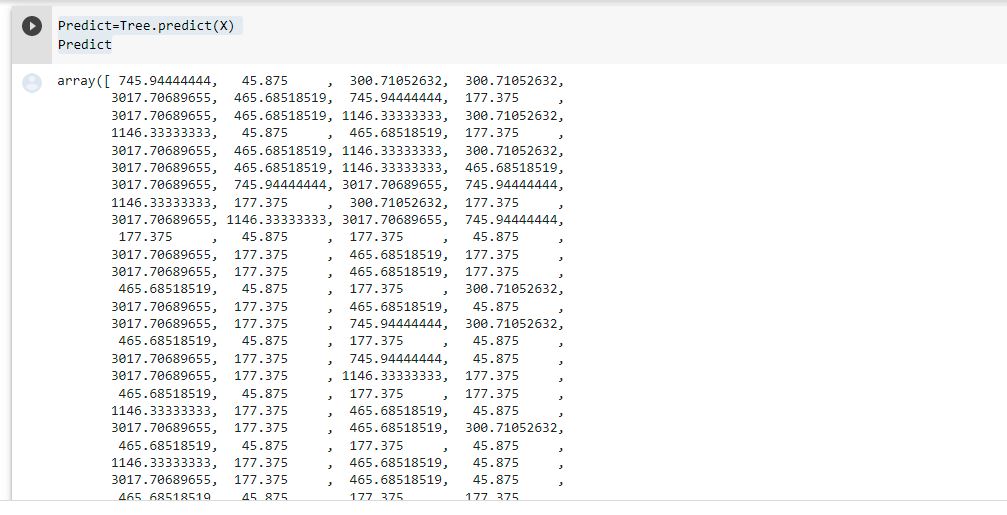
Tree=DecisionTreeRegressor(max\_depth=4,min\_samples\_leaf=0.1,random\_state=3)

Tree.fit(X,Y)



Predict=Tree.predict(X)

Predict



# RandomForestRegressor

from sklearn.ensemble import RandomForestRegressor

x= X.iloc [:, : -1]

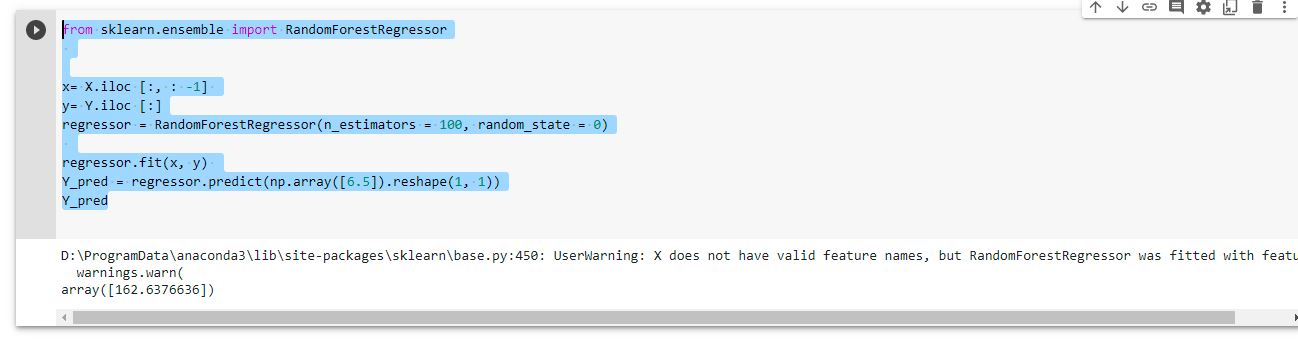
y= Y.iloc [:]

regressor = RandomForestRegressor(n\_estimators = 100, random\_state = 0)

regressor.fit(x, y)

Y\_pred = regressor.predict(np.array([6.5]).reshape(1, 1))

Y\_pred



# SVM

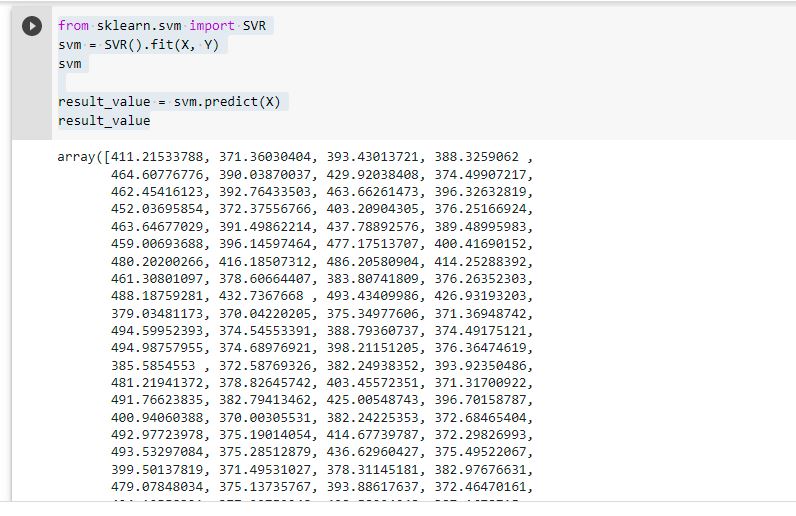
from sklearn.svm import SVR

svm = SVR().fit(X, Y)

svm

result\_value = svm.predict(X)

result\_value



# Confusion Matrix plot\_confusion\_matrix only supports classifiers So I am make an random data set

from sklearn.datasets import make\_classification

from sklearn.metrics import plot\_confusion\_matrix

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

from sklearn.svm import SVC

X, y = make\_classification(random\_state=0)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, random\_state=0)

clf = SVC(random\_state=0)

clf.fit(X\_train, y\_train)

SVC(random\_state=0)

plot\_confusion\_matrix(clf, X\_test, y\_test)

plt.show()



ROC AUC

An ROC curve shows the TPR as a function of FPR. Neither of these measures exists in the context of regression, so there is no such thing as ROC curves for regression.

# SVC AND LOGISTIC REGRESSION

from sklearn.metrics import roc\_curve, auc

from sklearn.svm import SVC

from sklearn.linear\_model import LogisticRegression

model\_logistic = LogisticRegression()

model\_logistic.fit(X\_train, y\_train)

y\_pred\_logistic = model\_logistic.decision\_function(X\_test)

model\_SVC = SVC(kernel = 'rbf', random\_state = 4)

model\_SVC.fit(X\_train, y\_train)

y\_pred\_svm = model\_SVC.decision\_function(X\_test)

logistic\_fpr, logistic\_tpr, threshold = roc\_curve(y\_test, y\_pred\_logistic)

auc\_logistic = auc(logistic\_fpr, logistic\_tpr)

svm\_fpr, svm\_tpr, threshold = roc\_curve(y\_test, y\_pred\_svm)

auc\_svm = auc(svm\_fpr, svm\_tpr)

plt.figure(figsize=(5, 5), dpi=100)

plt.plot(svm\_fpr, svm\_tpr, linestyle='-', label='SVM (auc = %0.3f)' % auc\_svm)

plt.plot(logistic\_fpr, logistic\_tpr, marker='.', label='Logistic (auc = %0.3f)' % auc\_logistic)

plt.xlabel('False Positive Rate -->')

plt.ylabel('True Positive Rate -->')

plt.legend()

plt.show()

