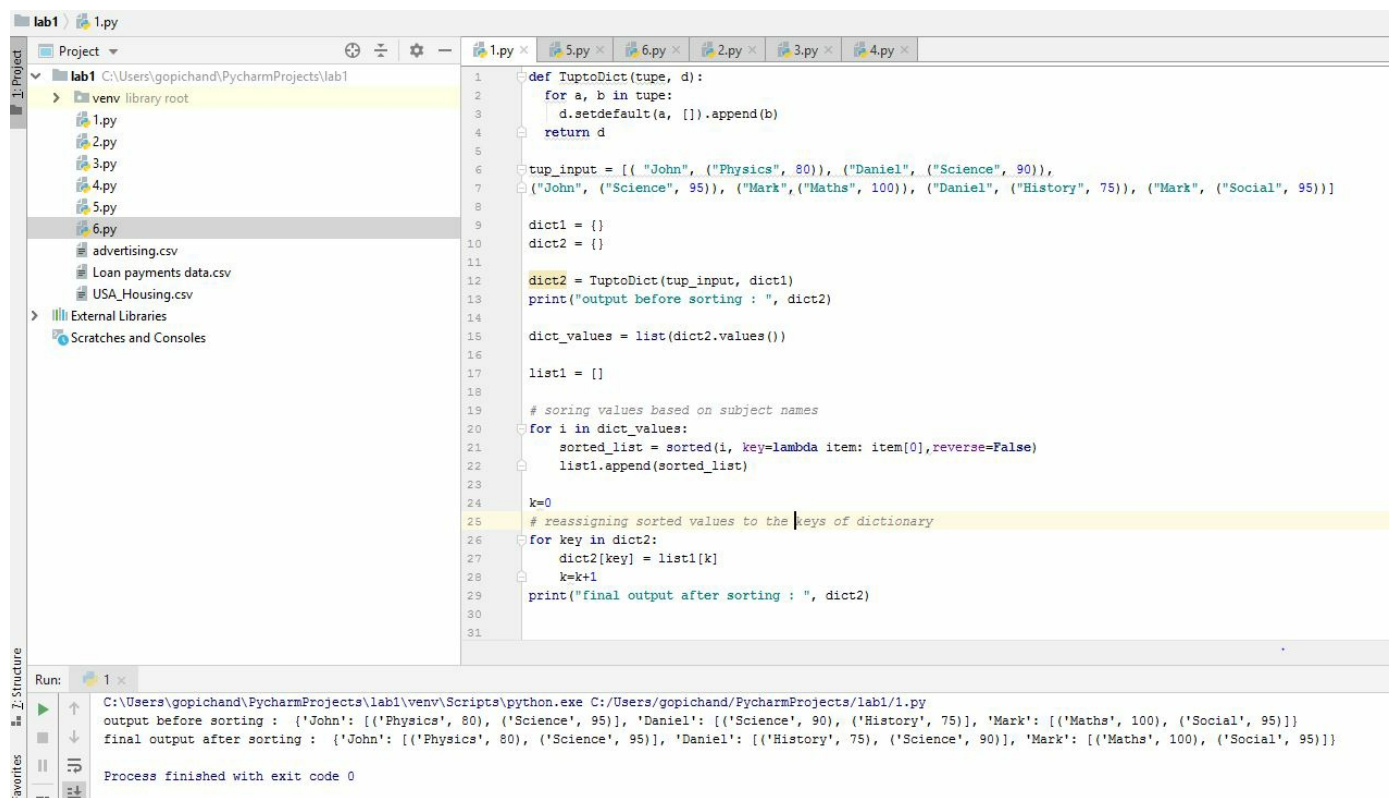


Welcome to the Python_Lab1 wiki team2!

1. In the first program we have taken list of tuples as input. A function was created and we have used setdefault inbuilt property to convert tuples to dictionary. After that we have sorted only the values and displayed the dictionary.



The screenshot shows the PyCharm IDE with a project named 'lab1'. The file explorer on the left shows a directory structure with files 1.py through 6.py and various CSV files. The main editor window displays the code for 1.py. The code defines a function 'TuptoDict' that takes a tuple and a dictionary, and uses 'setdefault' to append values to the dictionary. It then takes a list of tuples, converts them to a dictionary, sorts the values based on subject names, and reassigns the sorted values back to the dictionary keys. The output shows the dictionary before and after sorting.

```
1 def TuptoDict(tupe, d):
2     for a, b in tupe:
3         d.setdefault(a, []).append(b)
4     return d
5
6 tup_input = [ ("John", ("Physics", 80)), ("Daniel", ("Science", 90)),
7               ("John", ("Science", 95)), ("Mark", ("Maths", 100)), ("Daniel", ("History", 75)), ("Mark", ("Social", 95))]
8
9 dict1 = {}
10 dict2 = {}
11
12 dict2 = TuptoDict(tup_input, dict1)
13 print("output before sorting : ", dict2)
14
15 dict_values = list(dict2.values())
16
17 list1 = []
18
19 # sorting values based on subject names
20 for i in dict_values:
21     sorted_list = sorted(i, key=lambda item: item[0], reverse=False)
22     list1.append(sorted_list)
23
24 k=0
25 # reassigning sorted values to the keys of dictionary
26 for key in dict2:
27     dict2[key] = list1[k]
28     k=k+1
29 print("final output after sorting : ", dict2)
30
31
```

Run: 1 x

C:\Users\gopichand\PycharmProjects\lab1\venv\Scripts\python.exe C:\Users\gopichand\PycharmProjects\lab1\1.py

output before sorting : {'John': [('Physics', 80), ('Science', 95)], 'Daniel': [('Science', 90), ('History', 75)], 'Mark': [('Maths', 100), ('Social', 95)]}

final output after sorting : {'John': [('Physics', 80), ('Science', 95)], 'Daniel': [('History', 75), ('Science', 90)], 'Mark': [('Maths', 100), ('Social', 95)]}

Process finished with exit code 0

2. Input string is entered from Console. We have written a for loop which iterates over each letter in the string and finally creates a list of substrings. Finally we have displayed only the substrings with maximum length

```
1 def long_substrs(str1):
2     curr_list = [] #for holding current string
3     sub_string_list = [] #for holding all sub strings
4
5     # for generating list of substrings without repeating characters
6     for letter in str1:
7         if letter in curr_list:
8             sub_string_list.append(''.join(curr_list))
9             curr_pos = curr_list.index(letter) + 1
10            curr_list = curr_list[curr_pos:]
11            # print(curr_list)
12            curr_list += letter
13            sub_string_list.append(''.join(curr_list))
14            # print(sub_string_list)
15
16 # for displaying only substrings that has maximum length of all substrings
17 max_len = max(len(k) for k in sub_string_list)
18 for k in sub_string_list:
19     if len(k) == max_len:
20         print((k, len(k)))
21
22 str = input("Enter a String : ")
23 long_substrs(str)
24
25
```

long_substrs()

Run: 2 x

C:\Users\gopichand\AppData\Local\Programs\Python\Python37-32\python.exe C:/Users/gopichand/PycharmProjects/lab1/2.py

Enter a String : **pvwkev**

('wke', 3)

('kew', 3)

Process finished with exit code 0

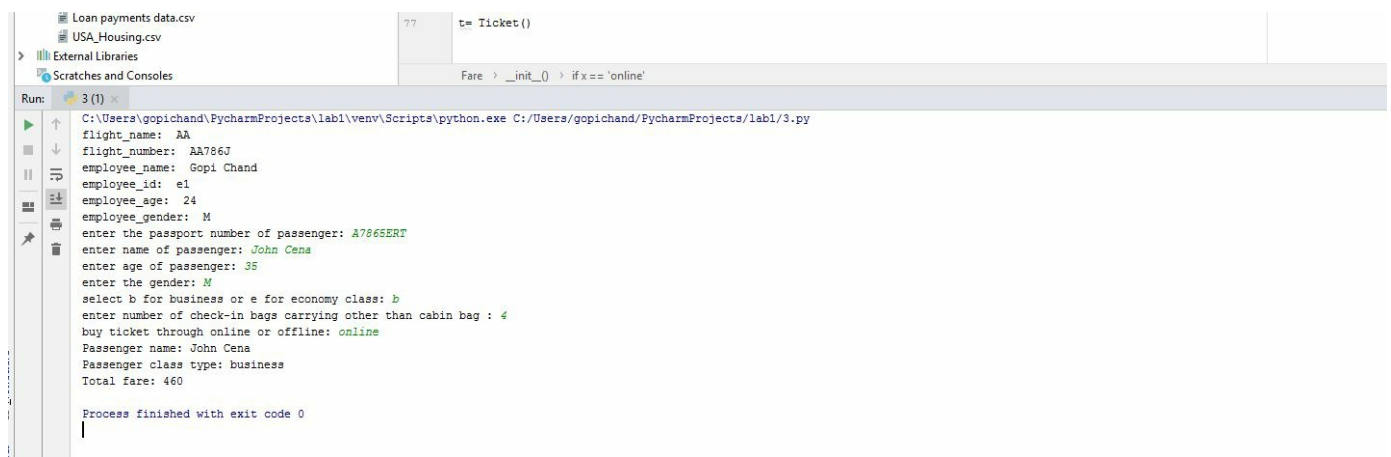
3. We have taken classes Flight, Employee, Passenger, Luggage, Fare and Ticket. Fare is inheriting from Luggage class and Ticket is inheriting from Passenger and Fare classes. There is super call in Fare class. There are also two private variables in Employee and Passenger classes respectively. It will take passenger details as input and give total fare of ticket based on type of purchase and number of check in bags

```
lab1 > 3.py
Project
lab1 C:\Users\gopichand\PycharmProjects\lab1
venv library root
1.py
2.py
3.py
4.py
5.py
6.py
advertising.csv
Loan payments data.csv
USA_Housing.csv
External Libraries
Scratches and Consoles

1
class Flight:
2
    def __init__(self, f_name, f_num):
3
        self.f_name = f_name
4
        self.f_num = f_num
5
6
    def flight_display(self):
7
        print('flight_name: ', self.f_name)
8
        print('flight_number: ', self.f_num)
9
10
class Employee:
11
    def __init__(self, e_id, e_name, e_age, e_gender):
12
        self.e_name = e_name
13
        self.e_age = e_age
14
        self.e_id = e_id #private variable
15
        self.e_gender = e_gender
16
    def emp_display(self):
17
        print("employee name: ", self.e_name)
18
        print('employee_id: ', self.e_id)
19
        print('employee_age: ', self.e_age)
20
        print('employee_gender: ', self.e_gender)
21
22
class Passenger:
23
    def __init__(self):
24
        Passenger._passport_number = input("enter the passport number of passenger: ") #private variable
25
        Passenger.name = input('enter name of passenger: ')
26
        Passenger.age = int(input('enter age of passenger: '))
27
        Passenger.gender = input('enter the gender: ')
28
        Passenger.class_type = input('select b for business or e for economy class: ')
29
30
class Luggage():
31
    bag_fare = 0
32
    def __init__(self, checkin_bags):
33
        self.checkin_bags = checkin_bags
34
        if checkin_bags > 2 :
35
            self.bag_fare = (checkin_bags-2)*40
36
        else:
37
            pass
38
39
class Fare(Luggage): #inheritance
    Employee > init ()
```

```
lab1 > 3.py
Project
lab1 C:\Users\gopichand\PycharmProjects\lab1
venv library root
1.py
2.py
3.py
4.py
5.py
6.py
advertising.csv
Loan payments data.csv
USA_Housing.csv
External Libraries
Scratches and Consoles

39
class Fare(Luggage): #inheritance
40
    offline = 250
41
    online = 300
42
    total_fare=0
43
    def __init__(self):
44
        super().__init__(int(input('enter number of check-in bags carrying other than cabin bag : '))) #super call
45
        x = input('buy ticket through online or offline: ')
46
        if x == 'online':
47
            Fare.total_fare = self.online + self.bag_fare
48
        elif x == 'offline':
49
            Fare.total_fare = self.offline + self.bag_fare
50
        else:
51
            pass
52
53
class Ticket(Passenger, Fare): #inheritance
54
    def __init__(self):
55
        print("Passenger name:", Passenger.name)
56
        if Passenger.class_type == "b":
57
            str = "business"
58
            Fare.total_fare+=80
59
        else:
60
            str = "economy"
61
            pass
62
        print("Passenger class type:", str)
63
        print("Total fare:", Fare.total_fare)
64
65
66
f1=Flight('AA', 'AA786J')
67
f1.flight_display()
68
69
emp1 = Employee('e1', 'Gopi Chand', 24, 'M')
70
emp1.emp_display()
71
72
p1 = Passenger()
73
74
fare1=Fare()
75
76
t= Ticket()
77
```



```
Loan payments data.csv
USA_Housing.csv
External Libraries
Scratches and Consoles
Fare > __init__() > if x == 'online'

Run: 3 (1) x
C:\Users\gopichand\PycharmProjects\lab1\venv\Scripts\python.exe C:/Users/gopichand/PycharmProjects/lab1/3.py
flight_name: AA
flight_number: AA786J
employee_name: Gopi Chand
employee_id: e1
employee_age: 24
employee_gender: M
enter the passport number of passenger: A786SERT
enter name of passenger: John Cena
enter age of passenger: 35
enter the gender: M
select b for business or e for economy class: b
enter number of check-in bags carrying other than cabin bag : 4
buy ticket through online or offline: online
Passenger name: John Cena
Passenger class type: business
Total fare: 460

Process finished with exit code 0
```

4. We have taken Advertisements dataset where 'Clicked on AD' column is the target column(number of hits on AD) depends on other features data like Area Income,Daily Internet Usage,Ad Topic Line,City,Country etc. We have done EDA by converting non-numerical features to numerical features and removed null values. We applied Multiple regression model to the data by taking all features once and taking only top correlated features once. By looking at r2 and rmse score, we observed that the model under performed for correlated data

```

4.py × 6.py × 5.py × 6th.py ×
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4
5 advv = pd.read_csv('advertising.csv')
6
7 print(advv.columns)
8
9 #converting non-numeric features to numeric features
10 advv['Country'] =advv['Country'].astype('category').cat.codes
11 advv['Ad Topic Line'] =advv['Ad Topic Line'].astype('category').cat.codes
12 advv['City'] =advv['City'].astype('category').cat.codes
13
14 #checking for nulls in the data
15 nulls = pd.DataFrame(advv.isnull().sum().sort_values(ascending=False))
16 nulls.columns = ['Null Count']
17 nulls.index.name = 'Feature'
18 # print(nulls)
19
20 #removing nulls in data
21 adv = advv.select_dtypes(include=[np.number]).interpolate().dropna()
22 print(sum(advs.isnull().sum() != 0))
23
24
25 print(adv.head())
26
27 df = pd.DataFrame(adv)
28
29 # checking correlation of all the columns against target column
30 print(' correlation of all columns are :\n' + str(df[df.columns:].corr()['Clicked on Ad'].sort_values(ascending=False)))
31
32 # Taking all columns for analysis
33 X = adv.drop('Clicked on Ad',axis=1)
34 y = adv['Clicked on Ad']
35
36 # taking top correlated columns for analysis
37 A = adv[['Age', 'Ad Topic Line', 'Country']]
38 b = adv['Clicked on Ad']
39

```



```
4.py x 6.py x 5.py x 6tth.py x
39
40 # splitting data into train data for training model and test data for testing model
41 from sklearn.model_selection import train_test_split
42 X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42, test_size=.2)
43 A_train, A_test, b_train, b_test = train_test_split(A, b, random_state=42, test_size=.2)
44
45 # fitting model to our train data
46 from sklearn.linear_model import LinearRegression
47 lm = LinearRegression()
48 lm.fit(X_train,y_train)
49
50 lm2 = LinearRegression()
51 lm2.fit(A_train,b_train)
52
53 # print(lm.intercept_)
54
55 coeff = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
56 # print(coeff)
57
58 # testing the model comparing with test data
59 predictions = lm.predict(X_test)
60 predictions1 = lm2.predict(A_test)
61 plt.scatter(y_test,predictions)
62 # plt.show()
63
64 # metrics
65 from sklearn.metrics import r2_score
66 print('total r2 score is ',r2_score(y_test,predictions))
67 print('correlated r2 score is ',r2_score(b_test,predictions1))
68 # print('r2 score is ',lm.score(X_test,y_test))
69 # print('r2 score is ',lm.score(y_test,predictions))
70 from sklearn.metrics import mean_squared_error
71 print('total rmse',mean_squared_error(y_test,predictions))
72 print('correlated rmse',mean_squared_error(b_test,predictions1))

run: 4 68.37 35 73889.99 ... 0 96 0
[5 rows x 9 columns]
correlation of all columns are :
Clicked on Ad 1.000000
Age 0.492531
Ad Topic Line 0.022787
Country 0.011415
City -0.007554
Male -0.038027
Area Income -0.476254
Daily Time Spent on Site -0.748117
Daily Internet Usage -0.786539
Name: Clicked on Ad, dtype: float64
total r2 score is 0.746225058354061
correlated r2 score is 0.25516818767147975
total rmse 0.06267606621300575
correlated rmse 0.18395483684983632
Process finished with exit code 0
```

5.We have taken Loan Payments data dataset where 'Paid Status' column is the target column(number of hits on AD) depends on other features data like Principal,terms,past_due_days,age,education,Gender etc. We have done EDA by converting non-numerical features to numerical features, removed null values, taken top correlated columns

for analysis. We applied 3 models namely Naive Bayes, SVM, KNN to the data and observed Naive Bayes is best model and followed by KNN and SVM

```
Python_Le 1 from sklearn.preprocessing import LabelEncoder
2 import pandas as pd
3 import numpy as np
4 df = pd.read_csv('Loan payments data.csv')
5 print(df.info())
6
7 #converting non-numeric features to numeric features using label encoder
8 le=LabelEncoder()
9 df['loan_status']=le.fit_transform(df['loan_status'])
10 df['Gender']=le.fit_transform(df['Gender'])
11 df['education']=le.fit_transform(df['education'])
12 df['past_due_days']=le.fit_transform(df['past_due_days'])
13
14 print(df.head())
15 #checking for nulls in the data
16 nulls = pd.DataFrame(df.isnull().sum().sort_values(ascending=False))
17 nulls.columns = ['Null Count']
18 nulls.index.name = 'Feature'
19 # print(nulls)
20
21 #removing nulls in data
22 df1 = df.select_dtypes(include=[np.number]).interpolate().dropna()
23 print(sum(df1.isnull().sum() != 0))
24
25 # checking correlation of all the columns against target column
26 print(' correlation of all columns are :\n' + str(df1[df1.columns[:]].corr()['loan_status'].sort_values(ascending=False)))
27
28 # taking top correlated columns for analysis
29 A = df1[['past_due_days', 'age', 'education']]
30 # taking target column
31 b = df1['loan_status']
32
33 # splitting data into train data for training model and test data for testing model
34 from sklearn.model_selection import train_test_split
35 A_train, A_test, b_train, b_test = train_test_split(A, b, random_state=42, test_size=.1)
36
37 # using naive bayes model
38 from sklearn.naive_bayes import GaussianNB
39 gnb = GaussianNB()
```

```

41 #using svm
42 from sklearn.svm import SVC
43 model = SVC()
44
45 #using knn
46 from sklearn.neighbors import KNeighborsClassifier
47 knn = KNeighborsClassifier()
48
49
50 # fitting model to our train data
51 gnb.fit(A_train,b_train)
52 model.fit(A_train,b_train)
53 knn.fit(A_train,b_train)
54
55 # testing the model comparing with test data
56 predictions_naive = gnb.predict(A_test)
57 predictions_svm = model.predict(A_test)
58 predictions_knn = knn.predict(A_test)
59
60
61 # calculating root mean square value of the errors
62 from sklearn.metrics import mean_squared_error
63 print('RMSE for naive bayes is: \n', mean_squared_error(b_test, predictions_naive))
64 print('RMSE for svm is: \n', mean_squared_error(b_test, predictions_svm))
65 print('RMSE for knn is: \n', mean_squared_error(b_test, predictions_knn))
66
67
68 from sklearn import metrics
69 print("Accuracy for naive bayes is : ",round(metrics.accuracy_score(b_test, predictions_naive) * 100, 2))
70 # print("classification_report\n",metrics.classification_report(y_test,predictions_naive))
71 # print("confusion matrix\n",metrics.confusion_matrix(y_test,predictions_naive))
72 print("Accuracy for svm is : ",round(metrics.accuracy_score(b_test, predictions_svm) * 100, 2))
73 # print("classification_report\n",metrics.classification_report(y_test,predictions_svm))
74 # print("confusion matrix\n",metrics.confusion_matrix(y_test,predictions_svm))
75
76 print("Accuracy for knn is : ",round(metrics.accuracy_score(b_test, predictions_knn) * 100, 2))
77 # print("classification_report\n",metrics.classification_report(y_test,predictions_knn))
78 # print("confusion matrix\n",metrics.confusion_matrix(y_test,predictions_knn))
79

```

```

Run: 5 x
[5 rows x 11 columns]
0
correlation of all columns are :
loan_status      1.000000
past_due_days    0.681331
age              0.032408
education        0.019917
Principal        -0.076873
Gender           -0.084500
terms           -0.098473
Name: loan_status, dtype: float64
C:\Users\gopichand\AppData\Local\Programs\Python\Python37-32\lib\site-packages\sklearn\svm\base.py:193: FutureWarning: The default value of gamma will change
"avoid this warning.", FutureWarning)
RMSE for naive bayes is:
0.02
RMSE for svm is:
0.3
RMSE for knn is:
0.04
Accuracy for naive bayes is : 98.0
Accuracy for svm is : 82.0
Accuracy for knn is : 96.0

Process finished with exit code 0

```

6. We have taken USA_Housing dataset where 'Price' column is the target column (number of hits on AD) depends on other features data like Avg. Area Income, Avg. Area House Age, Avg. Area Number of

Rooms,Avg. Area Number of Bedrooms,Area Population etc. We have done EDA by converting non-numerical features to numerical features, removed null values. After plotting the graphs of each column against Price, we observed Avg. Area Number of Rooms plot is best suited for this analysis. By seeing elbow graph, we can found k=3 is best. We have done visualizations and calculated silhouette score by taking different number of clusters from 2 to 9

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 from sklearn.decomposition import PCA
5 import seaborn as sns
6
7
8 data = pd.read_csv('USA_Housing.csv')
9 nulls = pd.DataFrame(data.isnull().sum().sort_values(ascending=False))
10 nulls.columns = ['Null Count']
11 nulls.index.name = 'Feature'
12 print(nulls)
13
14 print(data.columns)
15
16 # Index(['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',
17 #        'Avg. Area Number of Bedrooms', 'Area Population', 'Price', 'Address'],
18 #        dtype='object')
19
20 house = data.select_dtypes(include=[np.number]).interpolate().dropna()
21 print(sum(house.isnull().sum() != 0))
22 nulls = pd.DataFrame(house.isnull().sum().sort_values(ascending=False))
23 nulls.columns = ['Null Count']
24 nulls.index.name = 'Feature'
25 print(nulls)
26
27 print(' correlation of all columns are :\n' + str(data[data.columns[:]].corr()['Price'].sort_values(ascending=False)))
28
29 sns.FacetGrid(data,size=7)\
30 .map(plt.scatter,'Avg. Area Number of Rooms','Price')\
31 .add_legend()
32 plt.show()
33
34 x = house.iloc[:, [0,1,2,3,4,5]]
35 y = house.iloc[:, -1]
36 print(x.shape, y.shape)
37
38 from sklearn import preprocessing
39 scaler = preprocessing.StandardScaler()
```

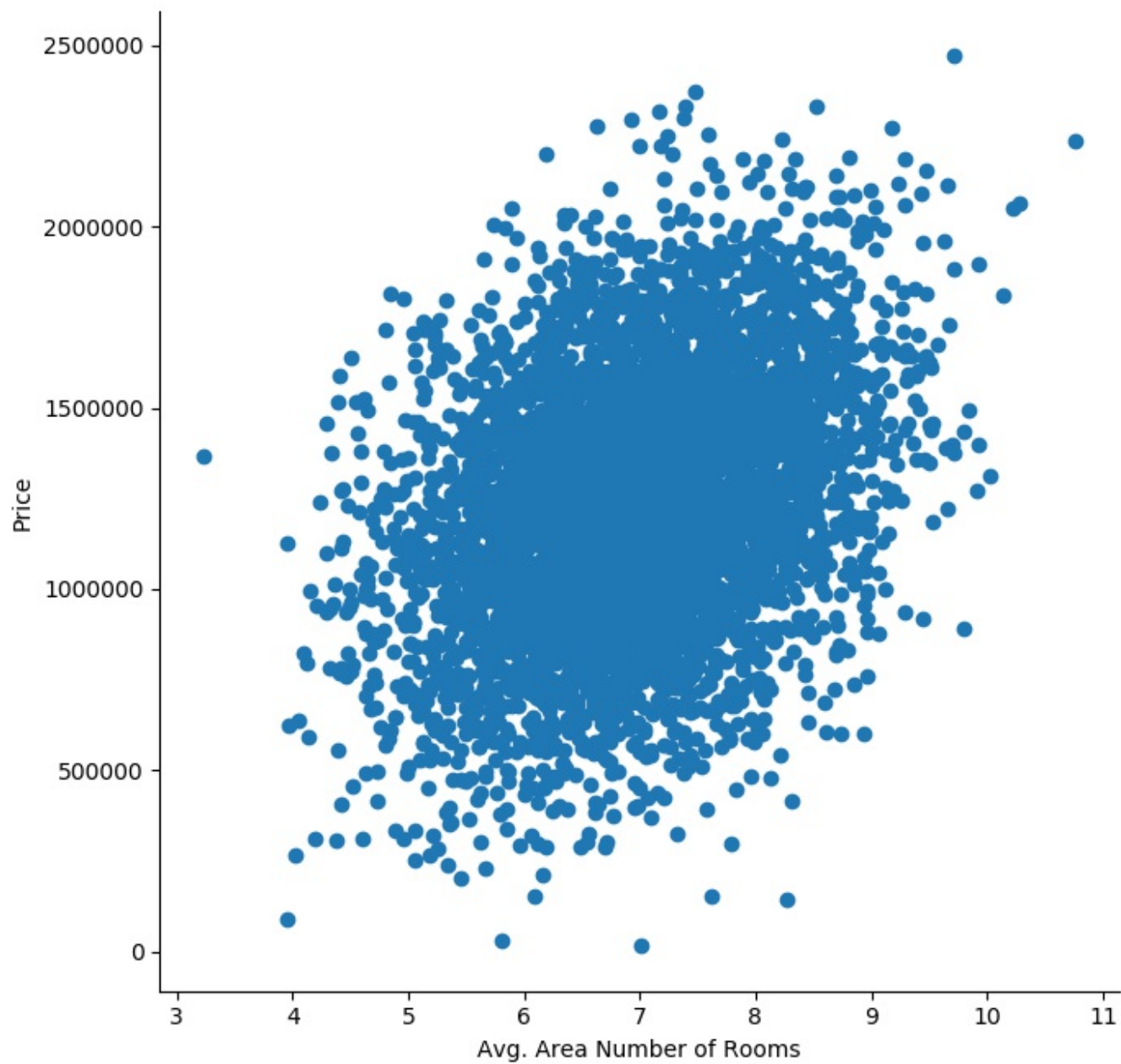
4.py ×
6.py ×
5.py ×
6thh.py ×

Le

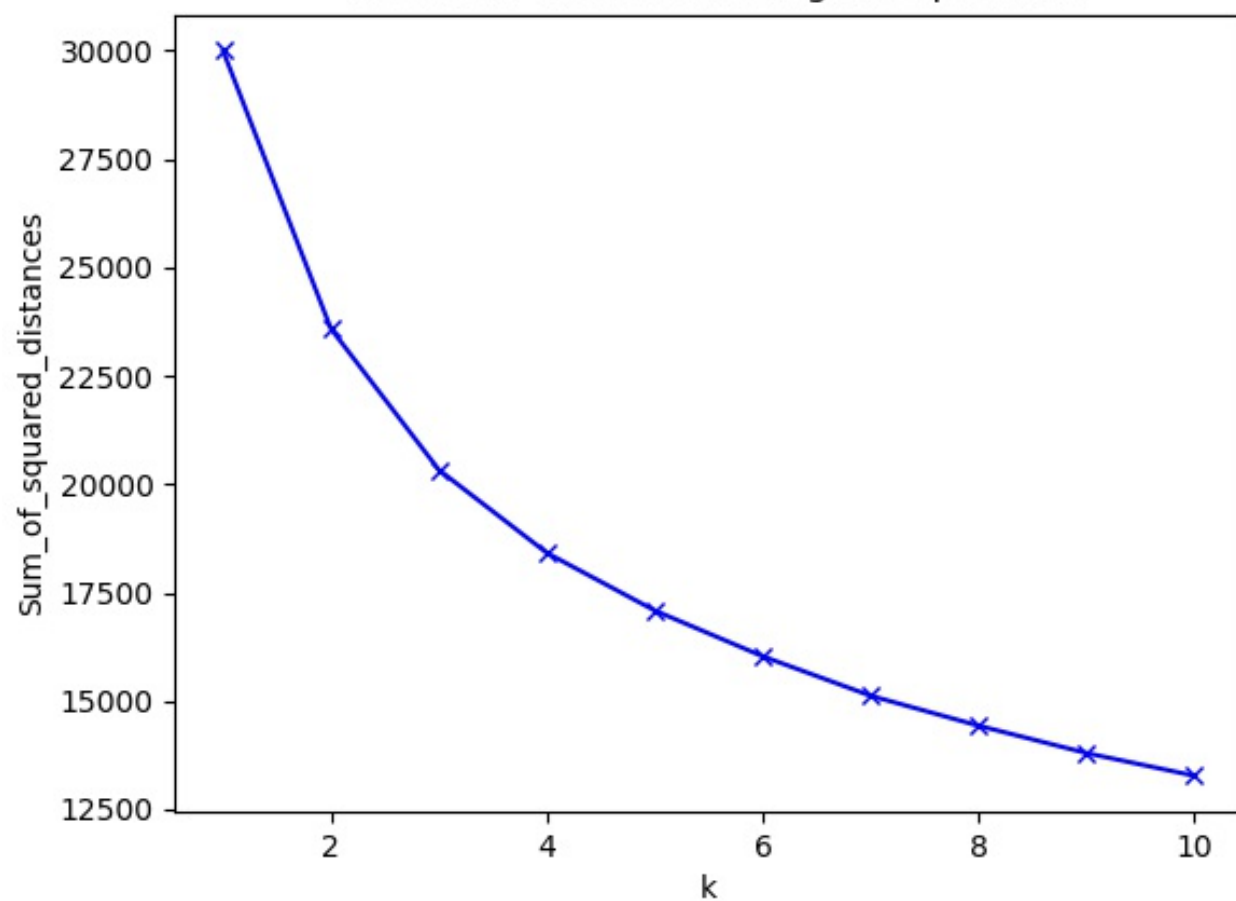
```

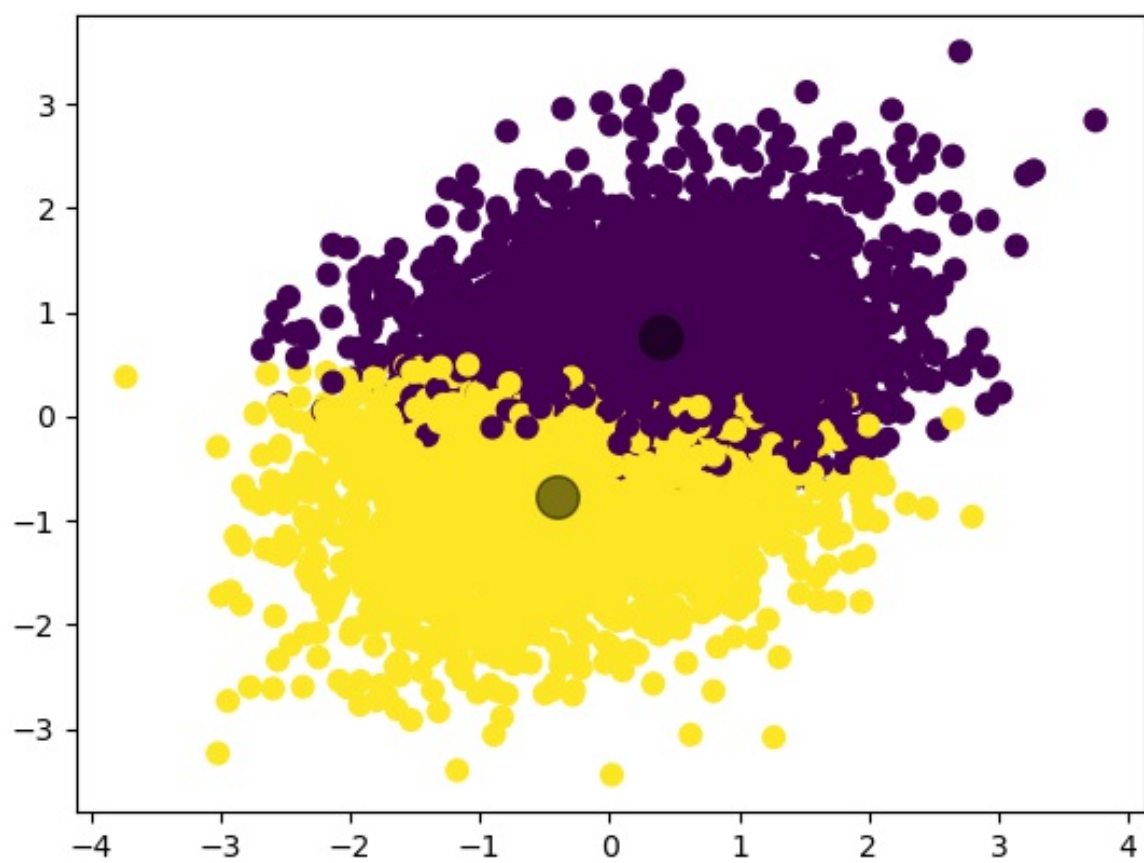
37
38 from sklearn import preprocessing
39 scaler = preprocessing.StandardScaler()
40 from sklearn import metrics
41 scaler.fit(x)
42 X_scaled_array = scaler.transform(x)
43 X_scaled = pd.DataFrame(X_scaled_array, columns = x.columns)
44
45 from sklearn.cluster import KMeans
46 Sum_of_squared_distances = []
47 K = range(1,11)
48 for k in K:
49     km = KMeans(n_clusters=k,init='k-means++',max_iter=300,n_init=10,random_state=0)
50     km = km.fit(X_scaled)
51     Sum_of_squared_distances.append(km.inertia_)
52
53
54 plt.plot(K, Sum_of_squared_distances, 'bx-')
55 plt.xlabel('k')
56 plt.ylabel('Sum_of_squared_distances')
57 plt.title('The Elbow Method showing the optimal k')
58 plt.show()
59
60
61 nclusters = 3 # this is the k in kmeans
62 seed = 0
63 K = range(2,10)
64 for k in K:
65     km = KMeans(n_clusters=k, random_state=seed)
66     km.fit(X_scaled)
67     # predict the cluster for each data point
68     y_cluster_kmeans = km.predict(X_scaled)
69     plt.scatter(X_scaled_array[:, 2], X_scaled_array[:, 5], c=y_cluster_kmeans, s=50)
70     centers = km.cluster_centers_
71     plt.scatter(centers[:, 2], centers[:, 5], c='black', s=200, alpha=0.5)
72     plt.show()
73     score = metrics.silhouette_score(X_scaled, y_cluster_kmeans)
74     print('silhouette score for clusters ' +str(k)+' is : ' + str(score))
75

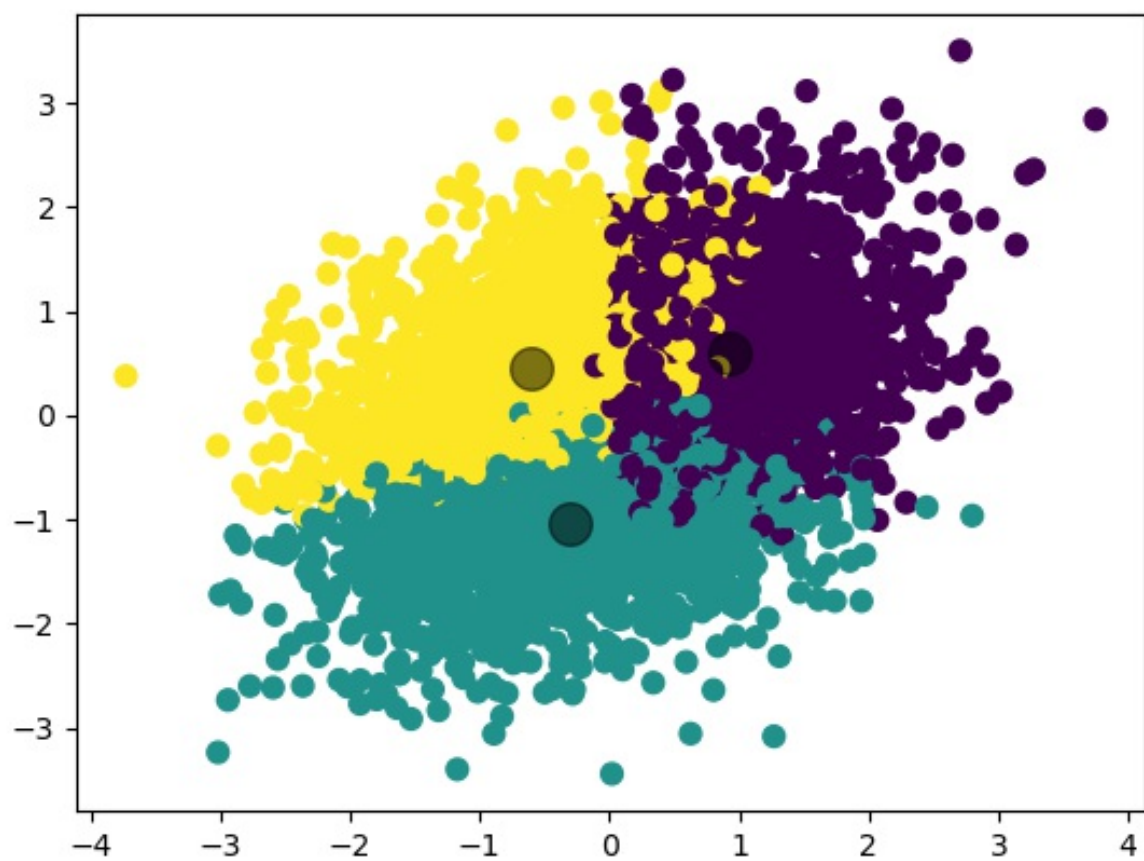
```

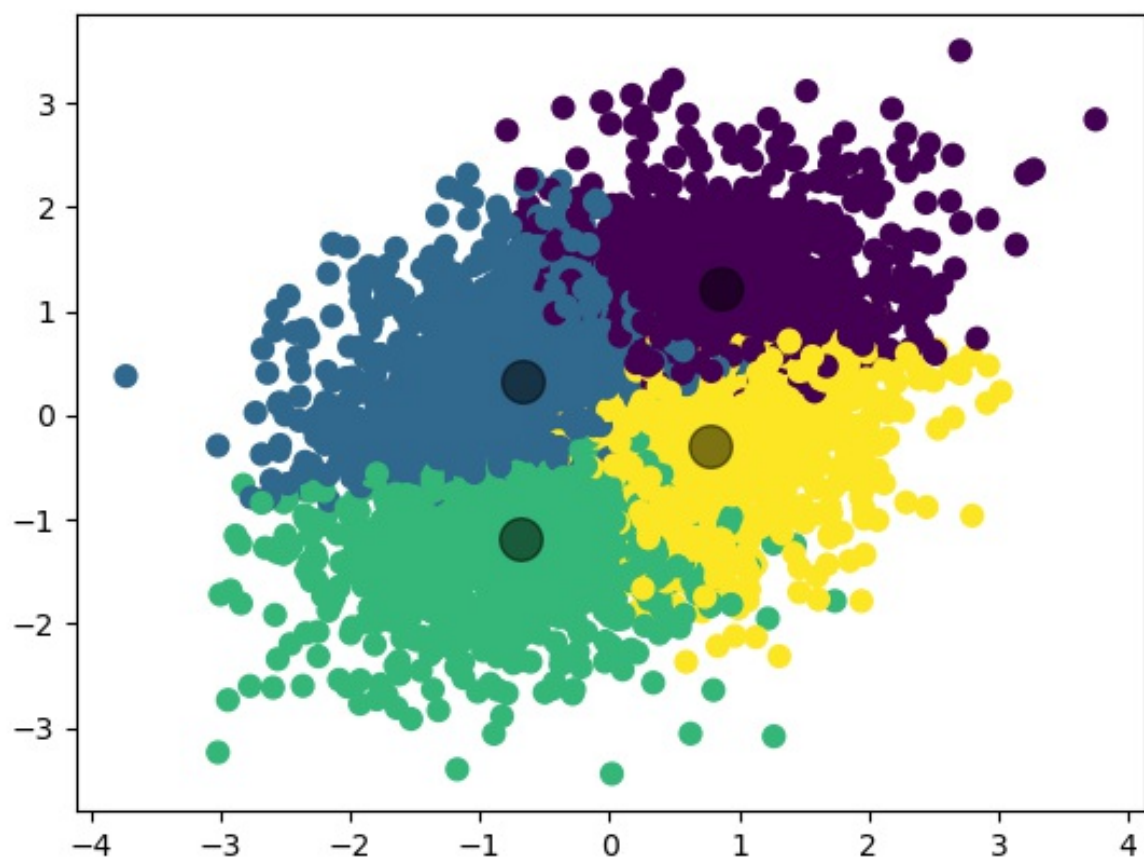


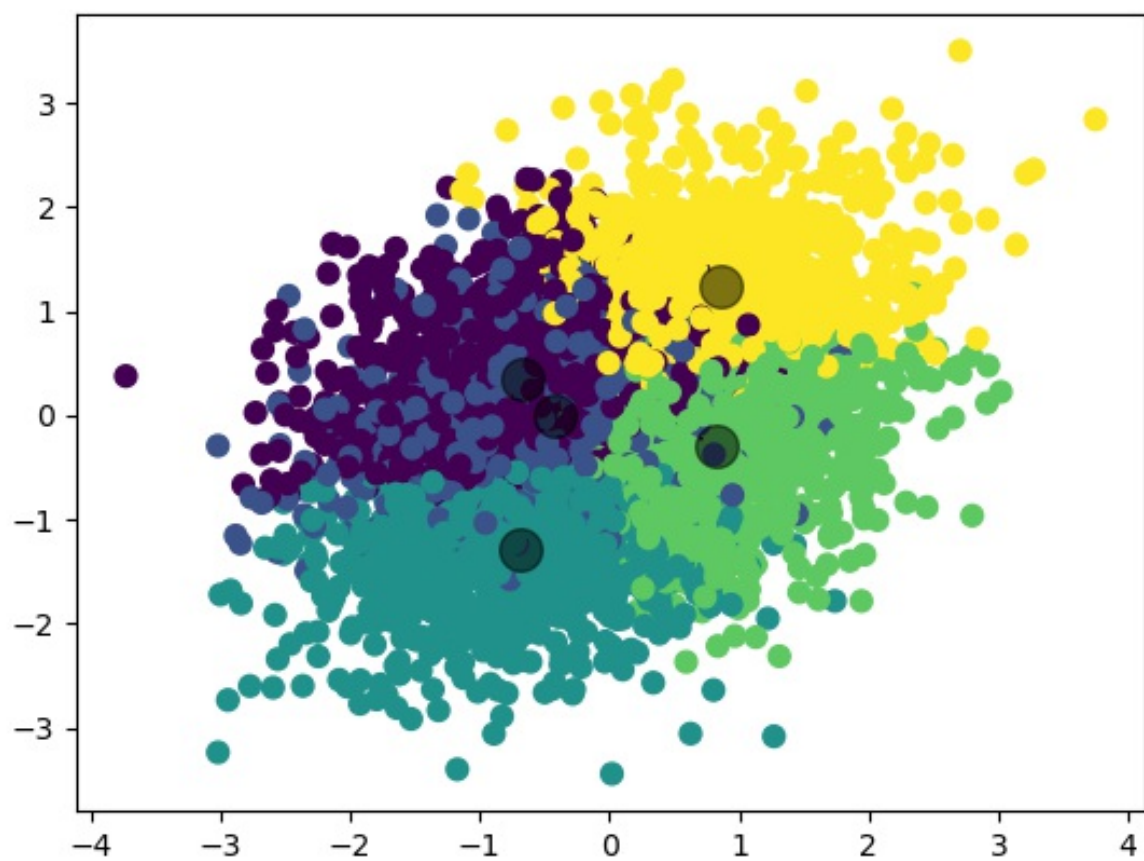
The Elbow Method showing the optimal k

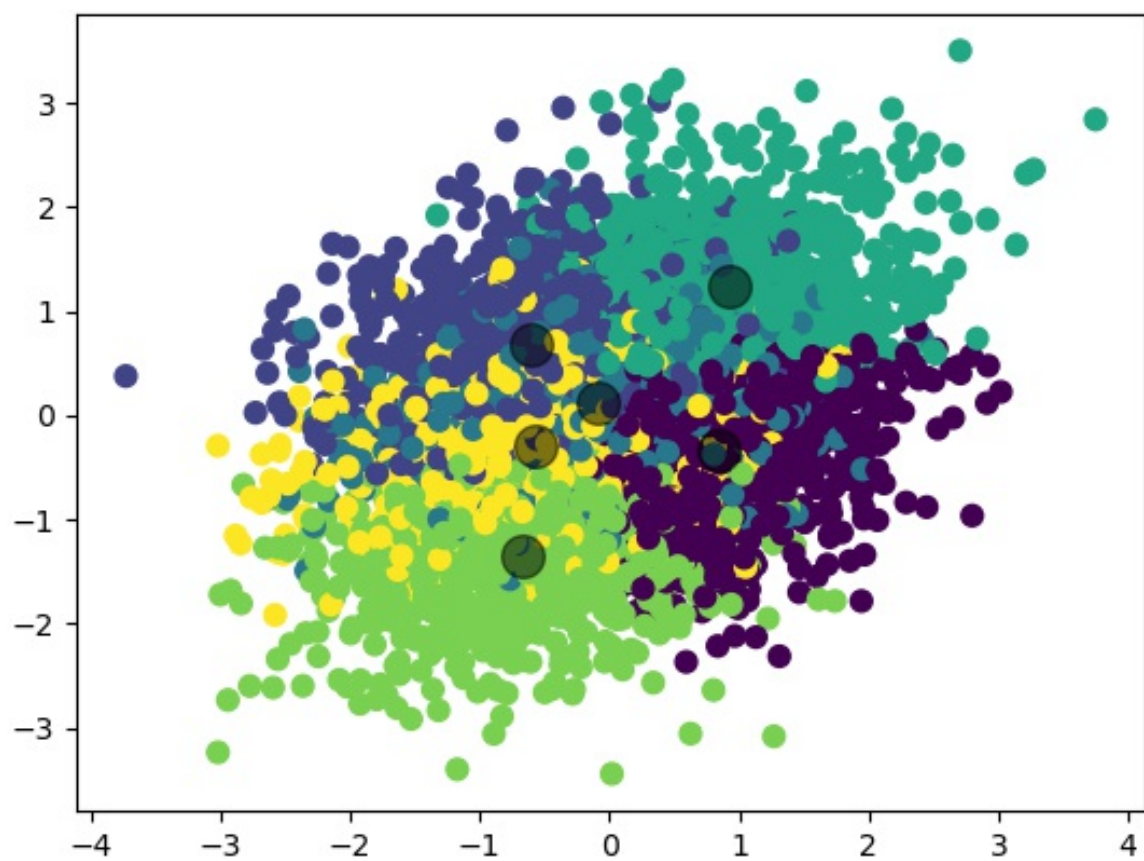


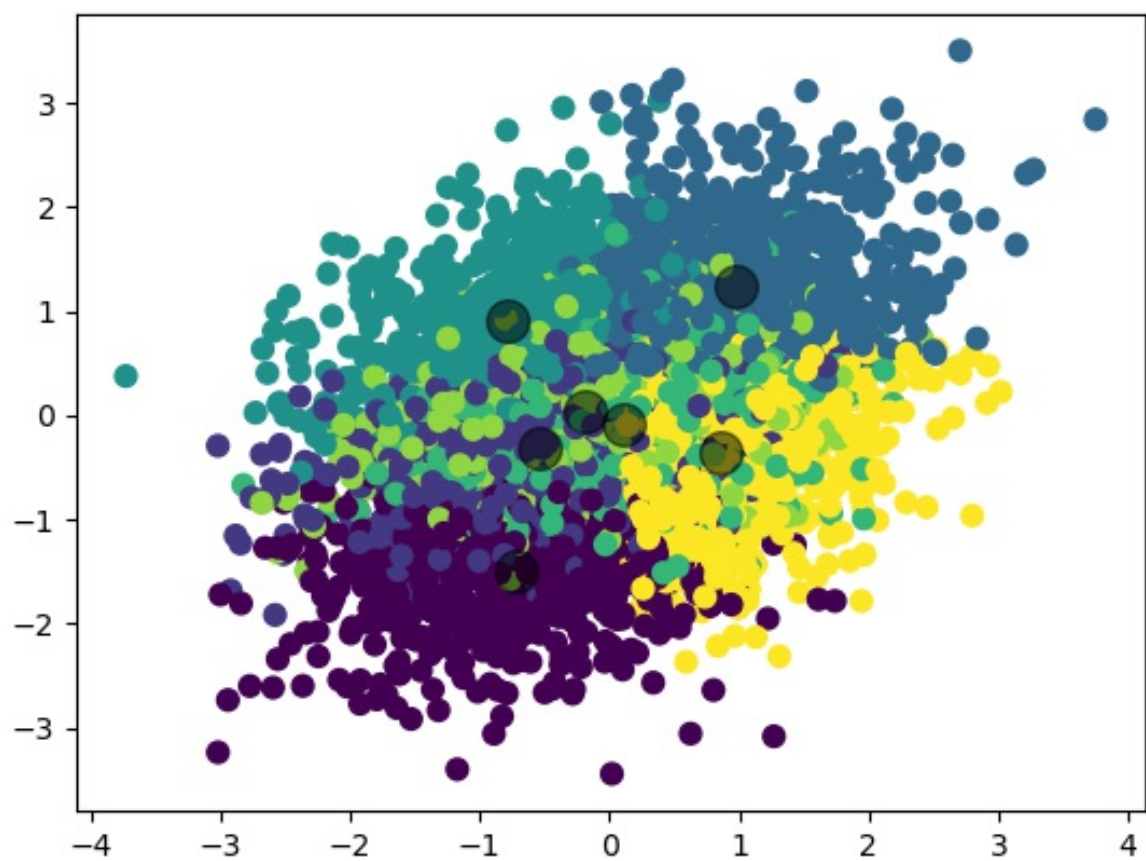


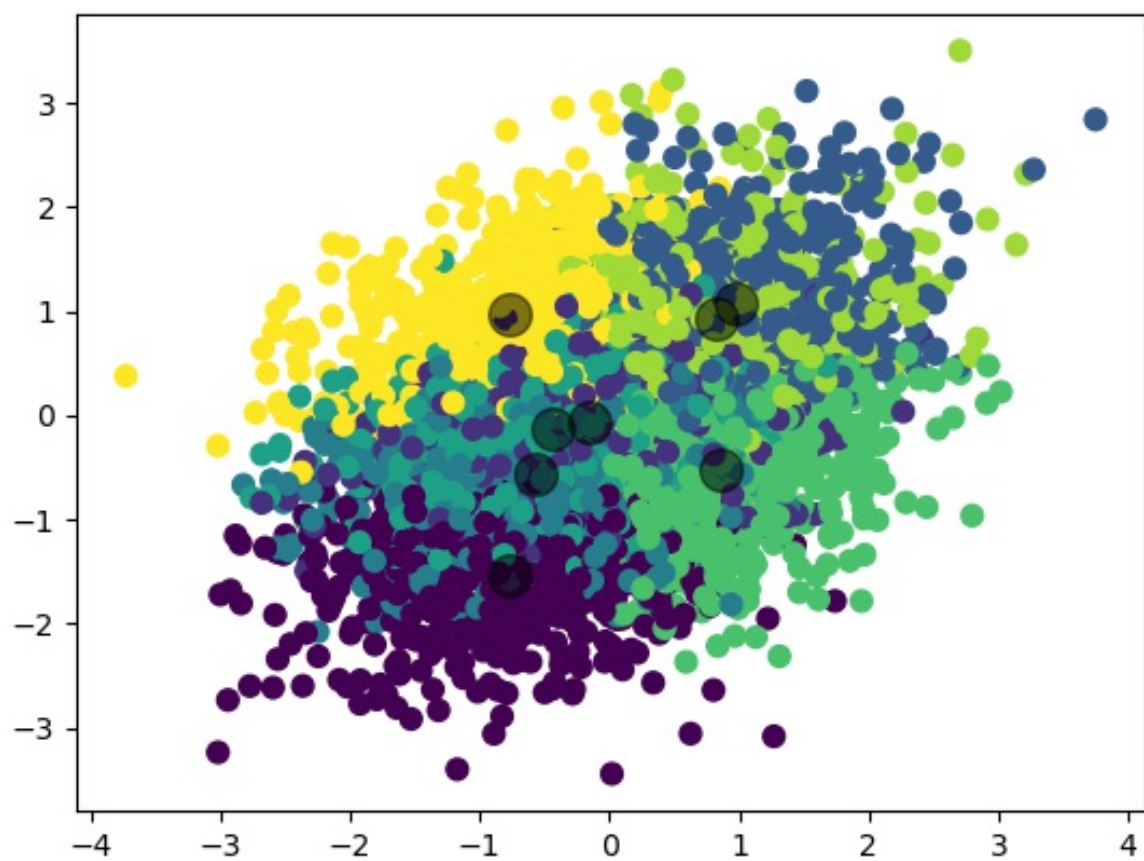


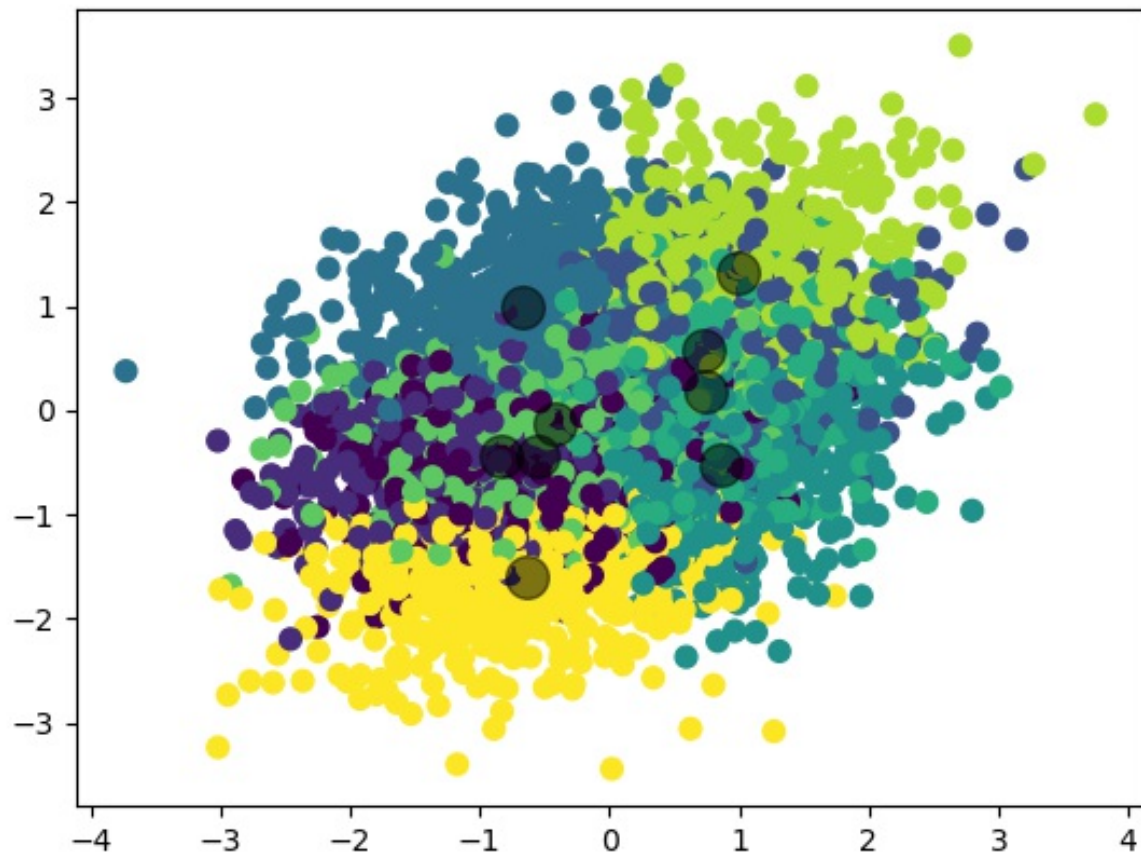












```
6th x
C:\Users\gopichand\appdata\local\Programs\Python\Python37-32\lib\site-packages\seaborn\axisgrid.py:30: UserWarning: The 'size' parameter has been renamed to 'height';
warnings.warn(msg, UserWarning)
(5000, 6) (5000,)
silhouette score for clusters 2 is : 0.19162386987794566
silhouette score for clusters 3 is : 0.1725904632708109
silhouette score for clusters 4 is : 0.159231305862676
silhouette score for clusters 5 is : 0.15495859919771884
silhouette score for clusters 6 is : 0.1504227088465428
silhouette score for clusters 7 is : 0.1509412651821385
silhouette score for clusters 8 is : 0.14689803096144988
silhouette score for clusters 9 is : 0.14420749380504008

Process finished with exit code 0
```