データの読み込み

In [42]:
```
import pandas as pd
from sklearn import metrics
from sklearn.neighbors import KNeighborsClassifier
from statistics import mean
import warnings

%matplotlib inline
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, cohen_kappa_score
```

In [18]:
```
zip_train = pd.read_csv("https://mjin.doshisha.ac.jp/data/zipTrain.csv")
zip_test = pd.read_csv("https://mjin.doshisha.ac.jp/data/zipTest.csv")
X_train = zip_train[zip_train.columns[zip_train.columns != "V1"]]
Y_train = zip_train[zip_train.columns[zip_train.columns == "V1"]]
X_test = zip_test[zip_test.columns[zip_test.columns != "V1"]]
Y_test = zip_test[zip_test.columns[zip_test.columns == "V1"]]
```

データの説明

学習データ：zip.train 7291行257列

テストデータ: zip.test　　2007行257

第１列が目的変数　　0, 1, 2, 3, 4, 5, 6, 7, 8, 9

問題1. knn法(knn{class})で学習データとテストデータを用いて、ｋ=2～5の正解率を計算するスクリプトを作成し、正解率が最も高いkを示せ。　計算時間は約2分かかる

In [43]:
```
warnings.simplefilter('ignore')

acc_list = [] # 各kにおける正解率を保存するリスト

for i in range(2, 6):
    knn = KNeighborsClassifier(n_neighbors = i) # knnのインスタンスを定義
    knn.fit(X_train, Y_train)
    Y_pred = knn.predict(X_test)
    acc = metrics.accuracy_score(Y_test, Y_pred) # 正解率の算出
    acc_list.append(acc)

df_acc = pd.DataFrame({"正解率": acc_list}, index = ["k=2", "k=3", "k=4", "k=5"])
df_acc
```

Out[43]:

	正解率
k=2	0.941206
k=3	0.944694
k=4	0.943199
k=5	0.944694

結果: k=3, 5のときが最も正解率が高い

問題2. 問題1の計算を10回繰り返すスクリプトを作成し、その平均値を用いてｋをいくらにした方がよいかを説明せよ。計算時間は約10分かかる。

In [44]:
```
k_acc_list = [] # それぞれのkにおける正解率を保存するリスト
acc_mean_list = [] # 各kにおける平均正解率を保存するリスト

for i in range(2, 6):
    for j in range(10):
        knn = KNeighborsClassifier(n_neighbors = i) # knnのインスタンスを定義
        knn.fit(X_train, Y_train)
        Y_pred = knn.predict(X_test)
        acc = metrics.accuracy_score(Y_test, Y_pred) # 正解率の算出
        k_acc_list.append(acc)
    acc_mean_list.append(mean(k_acc_list))
```

```
df_acc = pd.DataFrame({"正解率": acc_mean_list}, index = ["k=2", "k=3", "k=4", "k=5"])
df_acc
```

Out[44]:

	正解率
k=2	0.941206
k=3	0.942950
k=4	0.943033
k=5	0.943448

問題3. 上記の結果の箱ひげ図と平均プロットを作成し、考察せよ

In [45]:
```
# acc_scores = (k_acc_list[0:10], k_acc_list[10:20], k_acc_list[20:30], k_acc_list[30:40])

# # 箱ひげ図
# fig, ax = plt.subplots()

# bp = ax.boxplot(acc_scores) # 複数指定する場合はタプル型で渡します。
# ax.set_xticklabels(["k=2", "k=3", "k=4", "k=5"])

# plt.grid() # 横線ラインを入れることができます。
# # 描画
# plt.show()

# # plt.errorbar
```

問題4. 問題2で求めた最適のkを用いて計算を行い、テストデータに対する結果の正解率、kappa値、F値を求めよ。既存の関数を用いてもよい。

In [46]:
```
knn = KNeighborsClassifier(n_neighbors = 3) # knnのインスタンスを定義
knn.fit(X_train, Y_train)
Y_pred = knn.predict(X_test)
acc = metrics.accuracy_score(Y_test, Y_pred) # 正解率の算出
kappa = cohen_kappa_score(Y_test, Y_pred) # kappa値
f1 = metrics.f1_score(Y_test, Y_pred, average = "macro") # F値
print(f"テストデータに対する正解率は{round(acc, 3)}")
print(f"テストデータに対するkappa値は{round(kappa, 3)}")
print(f"テストデータに対するF値は{round(f1, 3)}")
```

```
テストデータに対する正解率は0.945
テストデータに対するkappa値は0.938
テストデータに対するF値は0.941
```