تمرین دوم درس داده کاوی

سوال ۷)

```
dataset = pd.read_csv(path)
```

با این دستور داده را می‌خوانیم.

```
X = dataset.iloc[:, 1:].values
```

```
y = dataset.iloc[:,0]
```

مقادیر ویژگی‌های مستقل را در x میریزیم و وابسته را در y می‌ریزیم.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10)
```

داده ها را به دو قسمت تست و ترین تقسیم می‌کنیم.

```
classifier = KNeighborsClassifier(n_neighbors=1)
```

```
classifier.fit(X_train, y_train)
```

داده ها را با الگوریتم KN1 اجرا میکنیم.

```
y_pred = classifier.predict(X_test)
```

yهای بدست آمده از اجرای الگوریتم روی داده‌های تست را بدست می آوریم

```
print(confusion_matrix(y_test, y_pred))
```

```
print(classification_report(y_test, y_pred))
```

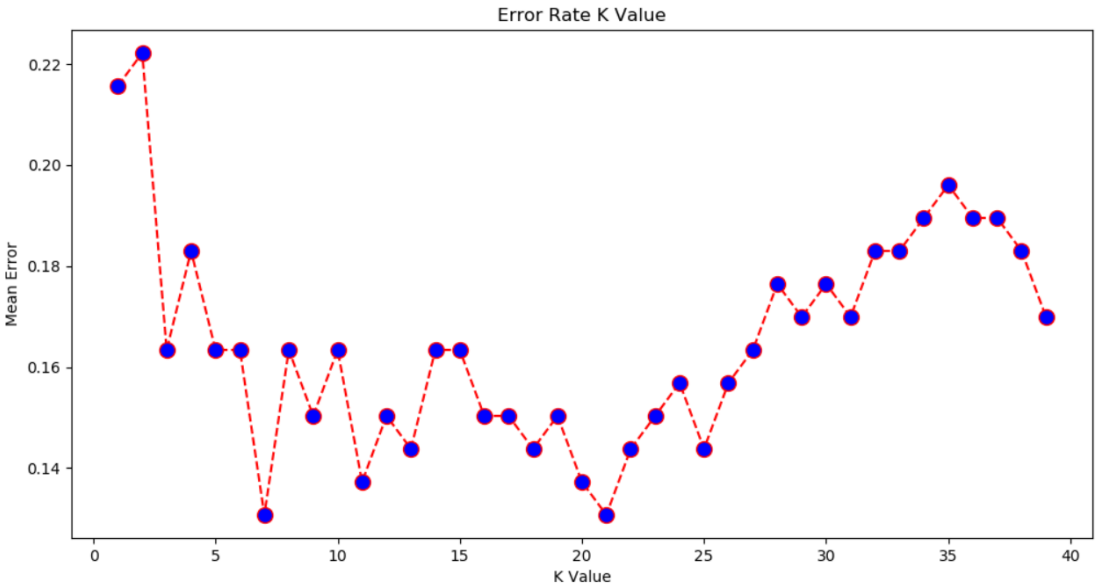
y مورد انتظار و بدست آمده را مقایسه می‌کنیم.

```
[[44 17]
 [21 71]]
              precision    recall  f1-score   support

           0       0.68      0.72      0.70        61
           1       0.81      0.77      0.79        92

   micro avg       0.75      0.75      0.75       153
   macro avg       0.74      0.75      0.74       153
weighted avg       0.76      0.75      0.75       153
```

در ادامه برای kهای بین ۱ تا ۴۰ این الگوریتم را اجرا میکنیم و ارور هر کدام را اندازه می‌گیریم.

```
error = []
```

```
for i in range(1, 40):
    knn = KNeighborsClassifier(n_neighbors=i)
    knn.fit(X_train, y_train)
    pred_i = knn.predict(X_test)
    error.append(np.mean(pred_i != y_test))

plt.figure(figsize=(12, 6))
plt.plot(range(1, 40), error, color='red', linestyle='dashed', marker='o',
        markerfacecolor='blue', markersize=10)
plt.title('Error Rate K Value')
plt.xlabel('K Value')
plt.ylabel('Mean Error')
plt.show()
```



سوال ۸) برای این سوال باید ابتدا مفادیر داده‌ها را به رقم تبدیل کنیم سپس می‌توان از کد نوشته شده استفاده کرد.

```
def importdata():
```

```
    train_data = pd.read_csv('venv/noisy_train.csv', sep=',', header=None)
    test_data = pd.read_csv('venv/noisy_test.csv',sep =',', header=None)
    valid_data = pd.read_csv('venv/noisy_valid.csv', sep=',', header=None)
    return train_data,test_data,valid_data
```

در این تابع شروع به خواندن ۳ فایل csv میکنیم.

```
def splitdataset(data_train,data_test):
    # Seperating the target variable
    X_train = data_train.values[1:, 1:]
    y_train = data_train.values[1:, 0]
    #print("y_train: ", y_train[:])

    X_test = data_test.values[1:, 1:]
    y_test = data_test.values[1:, 0]

    return X_train, X_test, y_train, y_test
```

دراین تابع ویژگیهای مستقل و وابسته داخل X_train,Y_train,X_test,Y_test قرار میگیرند.

```
def train_using_gini(X_train, X_test, y_train):
    # Creating the classifier object
    clf_gini = DecisionTreeClassifier(criterion="gini",
                                      random_state=100, max_depth=3, min_samples_leaf=5)

    clf_gini.fit(X_train, y_train)
    return clf_gini


def tarin_using_entropy(X_train, X_test, y_train):
    # Decision tree with entropy
```

```
    clf_entropy = DecisionTreeClassifier(
        criterion="entropy", random_state=100,
        max_depth=3, min_samples_leaf=5)

    # Performing training
    clf_entropy.fit(X_train, y_train)
    return clf_entropy


def prediction(X_test, clf_object):
    # Predicton on test with giniIndex
    y_pred = clf_object.predict(X_test)
    print("Predicted values:")
    print(y_pred)
    return y_pred


# Function to calculate accuracy
def cal_accuracy(y_test, y_pred):
    print("Confusion Matrix: ",
        confusion_matrix(y_test, y_pred))

    print("Accuracy : ",
        accuracy_score(y_test, y_pred) * 100)

    print("Report : ",
        classification_report(y_test, y_pred))
```

```
# Driver code
def main():
    # Building Phase
    data_train,data_test,data_valid = importdata()
    X_train, X_test, y_train, y_test = splitdataset(data_train,data_test)
    clf_gini = train_using_gini(X_train, X_test, y_train)
    clf_entropy = tarin_using_entropy(X_train, X_test, y_train)

    # Operational Phase
    print("Results Using Gini Index:")

    # Prediction using gini
    y_pred_gini = prediction(X_test, clf_gini)
    cal_accuracy(y_test, y_pred_gini)

    print("Results Using Entropy:")
    # Prediction using entropy
    y_pred_entropy = prediction(X_test, clf_entropy)
    cal_accuracy(y_test, y_pred_entropy)



if __name__ == "__main__":
    main()
```