Ishrana studenata

Tijana Jevtić
Jelena Mrdak

18. juni 2018

Sažetak

Koliko su bitne informacije o ishrani današnjim studentima? Da li njihove navike stečene još u detinjstvu utiču na to koju hranu danas vole da jedu? Koliko utiču njihove kulinarske veštine na način ishrane?

Na ova i još mnogobrojna pitanja, pokušaćemo da odgovorimo u ovom radu.

Sadržaj

1 Opis skupa podataka

Skup podataka uključuje informacije o izboru hrane, ishrani, preferencijama i ostalim informacijama dobijenim od studenata na koledžu. Postoji 126 odgovora. Podaci su ravni i nisu očišćeni.

- GPA
  numerički, prosek na fakultetu
- Pol
  kategorički
  1 - žensko
  2 - muško
- Doručak
  Ispitanicima je ponuđena slika pahuljica i krofne i treba da kažu šta ih asocira na doručak
  1 - pahuljice
  2 - krofna

- Procena kalorija u jednom parčetu piletine
  1 - 265
  2 - 430
  3 - 610
  3 - 720

- Da li je bitna količina kalorija koja se konzumira dnevno
  1 - ne znam koliko kalorija treba konzumirati dnevno
  2 - uopšte nije bitno
  3 - umereno je bitno
  3 - veoma je bitno

- Procena kalorija u scone from starbucks
  1 - 107 cal
  2 - 315 cal
  3 - 420 cal
  3 - 980 cal

- Kafa
  Ispitanicima su ponuđene dve slike i treba da kažu šta ih asocira na kafu.
  1 - creamy frapuccino
  2 - espreso

- Comfort food
  Ispitanici treba da navedu od 3 do 5 comfort food.

- comfort food reasons
  Ispitanici treba da navedu do 3 razloga zašto jedu comfort food? (npr. tuga, sreća, bes, itd)

- comfort food reasons coded
  1 - stres
  2 - dosada
  3 - depresija
  4 - glad
  5 - lenjost
  6 - hladno vreme
  7 - sreća
  8 - gledanje televizije
  9 - ništa od navedenog

2 Primer klasifikacije

Odredićemo klasifikaciju na osnovu atributa comfort_food_reasons_coded, cook i eating_out, dok će nam ciljni atribut biti weight.

Najpre ćemo učitati podatke i prikazati prvih pet redova.

```
df = pd.read_csv('./food-choices/food_coded.csv')
print('\n{}'.format(df.head()))
```

```
   GPA  Gender  breakfast ... waffle_calories                    weight
0   2.4       2          1 ...            1315                       187
1 3.654       1          1 ...             900                       155
2   3.3       1          1 ...             900  I'm not answering this.
3   3.2       1          1 ...            1315            Not sure, 240
4   3.5       1          1 ...             760                       190
```

Možemo uraditi osnovnu statistiku za svaku kolonu.

```
print("\nStatistike skupa:\n{}".format(df.describe()))
```

```
Statistike skupa:
           Gender   breakfast  calories_chicken ...  veggies_day    vitamins
count  125.000000  125.000000        125.000000 ...   125.000000  125.000000
mean     1.392000    1.112000        577.320000 ...     4.008000    1.512000
std      0.490161    0.316636        131.214156 ...     1.081337    0.501867
min      1.000000    1.000000        265.000000 ...     1.000000    1.000000
25%      1.000000    1.000000        430.000000 ...     3.000000    1.000000
50%      1.000000    1.000000        610.000000 ...     4.000000    2.000000
75%      2.000000    1.000000        720.000000 ...     5.000000    2.000000
max      2.000000    2.000000        720.000000 ...     5.000000    2.000000
```

Za algoritam koji želimo da primenimo, izdvojićemo sledeće atribute: comfort_food_reasons_coded, cook, eating_out i weight.

```
target_attribute = 'weight'
attribute_1 = 'comfort_food_reasons_coded'
attribute_2 = 'cook'
attribute_3 = 'eating_out'

df = df[[attribute_1, attribute_2, attribute_3, target_attribute]]
```

```
   comfort_food_reasons_coded  cook  eating_out                    weight
0                         9.0   2.0           3                       187
1                         1.0   3.0           2                       155
2                         1.0   1.0           2  I'm not answering this.
3                         2.0   2.0           2            Not sure, 240
4                         1.0   1.0           2                       190
```

Kao što možemo primetiti, nisu sve vrednosti celobrojne. Zato ćemo obrisati sve redove koji sadrže NaN-ove ili stringove u nekoj od ove četiri kolone. Takođe, vrednosti u koloni weight ćemo transformisati. Preslikaćemo ih u skup $\{0, 1, 2\}$.

```
df = df.replace('nan', np.nan)
df = df.dropna()


df = df[df[target_attribute].apply(lambda x: str(x).isdigit())]

df.reset_index(drop=True, inplace=True)

df[attribute_1] = df.comfort_food_reasons_coded.astype(int)
df[attribute_2] = df.cook.astype(int)
df[attribute_3] = df.eating_out.astype(int)
df[target_attribute] = df.weight.astype(int)

changes = {}
weight = df[target_attribute].unique()
for w in weight:
    if int(w) <150:
        changes[w] = 0
    elif int(w) < 190:
        changes[w] = 1
    else:
        changes[w] = 2

df[target_attribute] = df[target_attribute].replace(changes)
```

	comfort_food_reasons_coded	cook	eating_out	weight
0	9	2	3	1
1	1	3	2	1
2	1	1	2	2
3	4	3	1	2
4	1	2	2	1

Sada ćemo izvršiti podelu skupa na test i trening skup.

```
X = df[[attribute_1, attribute_2, attribute_3]]
y = df[[target_attribute]]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
print("\nVelicina skupa za obucavanje: {}".format(X_train.size))
print("Velicina skupa za testiranje: {}".format(X_test.size))
```

```
Velicina skupa za obucavanje: 207
Velicina skupa za testiranje: 90
```

Pošto smo izvršili podelu skupa, primenićemo algoritam za klasifikaciju - k najbližih suseda.

```
clf = KNeighborsClassifier(5, 'distance')

# Treniramo model
clf.fit(X_train, y_train.values.ravel())

# Vrsimo predikciju
y_test_predicted = clf.predict(X_test)
y_train_predicted = clf.predict(X_train)

# Izracunavamo preciznost
train_acc = clf.score(X_train, y_train)
test_acc = clf.score(X_test, y_test)
print('train preciznost: {}'.format(train_acc))
print('test preciznost: {}'.format(test_acc))
```

```
train preciznost: 0.7536231884057971
test preciznost: 0.7
```

Izveštaj i matricu konfuzije možemo dobiti na sledeći način:

```
test_rep = sklearn.metrics.classification_report(y_test, y_test_predicted)
train_rep = sklearn.metrics.classification_report(y_train, y_train_predicted)
print("\nTest izvestaj:\n{}".format(test_rep))
print("Trening izvestaj:\n{}".format(train_rep))

train_conf = sklearn.metrics.confusion_matrix(y_train, y_train_predicted)
test_conf = sklearn.metrics.confusion_matrix(y_test, y_test_predicted)
print("Matrica konfuzije za skup za obucavanje:\n{}".format(train_conf))
print("\nMatrica konfuzije za skup za testiranje:\n{}".format(test_conf))
```

```
Test izvestaj:
             precision    recall  f1-score   support

          0       0.67      0.91      0.77        11
          1       0.77      0.67      0.71        15
          2       0.50      0.25      0.33         4

avg / total       0.70      0.70      0.68        30

Trening izvestaj:
             precision    recall  f1-score   support

          0       0.74      0.86      0.79        29
          1       0.73      0.81      0.77        27
          2       1.00      0.38      0.56        13

avg / total       0.78      0.75      0.74        69
```

```
Matrica konfuzije za skup za obucavanje:
[[25  4  0]
 [ 5 22  0]
 [ 4  4  5]]

Matrica konfuzije za skup za testiranje:
[[10  1  0]
 [ 4 10  1]
 [ 1  2  1]]
```

Literatura

[1] Knjiga1

[2] Knjiga2