

# Analiza skupa podataka Food choices

Tijana Jevtić  
Jelena Mrdak

26. juni 2018

## Sažetak

Skup podataka Food choices predstavlja skup odgovora studenata sa koledza na određena pitanja vezana za njihove navike u ishrani. Dakle, podaci su dobijeni anketiranjem 125 studenata na 61 pitanje različitog tipa. Neka pitanja se konkretno tiču njihovih preferenca vezanih za hranu, njihovih navika, dok se kroz ostala dobija više informacija o njihovom socio-ekonomskom statusu. Sve to zajedno čini bogat skup podataka iz kog se može zaključiti dosta zanimljivih pravilnosti.

Šta današnji studenti na koledžu najviše vole da jedu? Koliko često kuvaju? Zasto jedu i da li je jedini odgovor zbog gladi? Koliko to što se prejedaju i razlozi zbog kojih konzumiraju svoju omiljenu hranu utiču na njihovu težinu?

Na ova i još mnogobrojna pitanja, pokušaćemo da odgovorimo u ovom radu koristeći programski jezik python i adekvatne biblioteke.

# Sadržaj

1	Opis skupa podataka	4
2	Detaljnije upoznavanje sa skupom podataka	9
3	Primer klasifikacije	12
4	Vizuelizacija podataka	16
4.1	PCA . . . . .	18

# 1 Opis skupa podataka

Za početak ćemo se površno upoznati sa skupom podataka koji se nalazi u datoteci `food_coded.csv`.

```
df = pd.read_csv('./food_coded.csv', na_values="none")
print(data.shape)
```

Skup podataka je učitao, nedostajuće vrednosti su obeležene sa `none` i izlaz je:

```
(125, 61)
```

Skup se sastoji od 61 atributa i 125 redova.

Atribute dobijamo:

```
print(data.columns)
```

Spisak svih atributa:

```
Index(['GPA', 'Gender', 'breakfast', 'calories_chicken', 'calories_day',
      'calories_scone', 'coffee', 'comfort_food', 'comfort_food_reasons',
      'comfort_food_reasons_coded', 'cook', 'comfort_food_reasons_coded.1',
      'cuisine', 'diet_current', 'diet_current_coded', 'drink',
      'eating_changes', 'eating_changes_coded', 'eating_changes_coded1',
      'eating_out', 'employment', 'ethnic_food', 'exercise',
      'father_education', 'father_profession', 'fav_cuisine',
      'fav_cuisine_coded', 'fav_food', 'food_childhood', 'fries', 'fruit_day',
      'grade_level', 'greek_food', 'healthy_feeling', 'healthy_meal',
      'ideal_diet', 'ideal_diet_coded', 'income', 'indian_food',
      'italian_food', 'life_rewarding', 'marital_status',
      'meals_dinner_friend', 'mother_education', 'mother_profession',
      'nutritional_check', 'on_off_campus', 'parents_cook', 'pay_meal_out',
      'persian_food', 'self_perception_weight', 'soup', 'sports', 'thai_food',
      'tortilla_calories', 'turkey_calories', 'type_sports', 'veggies_day',
      'vitamins', 'waffle_calories', 'weight'],
      dtype='object')
```

U nastavku ćemo pomenuti svaki od atributa kako bi se čitalac upoznao sa skupom podataka, a posebnu pažnju ćemo posvetiti onima koje koristimo u istraživanju.

- GPA - numerički, prosek na fakultetu
- Pol - kategorički
  - 1 - žensko
  - 2 - muško
- Doručak - koja od 2 ponudjene slike ispitanike više asocira na doručak

- Procena kalorija u jednom parčetu piletine
  - 1 - 265
  - 2 - 430
  - 3 - 610
  - 4 - 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
  - 4 - veoma je bitno
- Procena kalorija u kolaču iz Starbucks-a
  - 1 - 107 cal
  - 2 - 315 cal
  - 3 - 420 cal
  - 4 - 980 cal
- Kafa - koja od 2 ponuđene slike ispitanike više asocira na kafu
- Hrana za utehu - koja hrana asocira ispitanike na dom i lepe uspomene, čini ih srećnim Ispitanici treba da navedu između 3 i 5 različitih jela.
- Zašto posežu za hranom za utehu  
Ispitanici treba da navedu do 3 razloga zašto jedu hranu za utehu (npr. tuga, sreća, bes)
- Hrana za utehu - kodirana
  - 1 - stres
  - 2 - dosada
  - 3 - depresija
  - 4 - glad
  - 5 - lenjost
  - 6 - hladno vreme
  - 7 - sreća
  - 8 - gledanje televizije
  - 9 - ništa od navedenog
- Kuvanje - koliko ispitanici često kuvaju
  - 1 - svakog dana
  - 2 - nekoliko dana nedeljno
  - 3 - koliko često mogu, ali retko
  - 4 - jedino za praznike
  - 5 - nikada
- Koji tip kuhinje su jeli kad su bili mali
  - 1 - američka
  - 2 - meksička/španska

- 3 - korejska/azijska
- 4 - indijska
- 5 - američka sa internacionalnim jelima
- 6 - ostalo

- Trenutna dijeta
- Trenutna dijeta - kodirano od 1 do 4
- Koju sliku asociraju sa picem
- Kako se ishrana promenila od kada su na koledzu
- Kako se ishrana promenila
  - 1 - na lošije
  - 2 - na bolje
  - 3 - isto
  - 4 - nije sigurno
- Kako se ishrana promenila u terminima toga šta jedu
- Prejedanje - koliko se ispitanici često prejedaju tokom nedelje
  - 1 - nikad
  - 2 - 1 do 2 puta
  - 3 - 2 do 3 puta
  - 4 - 3 do 5 puta
  - 5 - svaki dan
- Zaposlenje - da li su zaposleni 1 - da, puno radno vreme
  - 2 - da, poluradno vreme
  - 3 - ne
  - 4 - drugo
- Lokalna hrana - koliko je verovatno da bi jeli hranu sa nekom područja
- Vežbanje - koliko puta nedeljno vežbaju
  - 1 - svakodnevno
  - 2 - 2 do 3 puta
  - 3 - jednom
  - 4 - ponekad
  - 5 - nikad
- Očevo obrazovanje
- Očeva profesija
- Omiljena vrsta kuhinje
- Omiljena vrsta kuhinje - kodirano

- Omiljena hrana - kuvana ili kupljena
  - 1 - kuvana kod kuće
  - 2 - kupljena u prodavnici
  - 3 - oba
- Omiljena hrana iz detinjstva
- Koja od 2 slike ih više asocira sa krompirićima
- Voće - koliko je verovatno da će pojesti voće tokom dana
  - 1 - vrlo verovatno
  - 2 - ne toliko verovatno
  - 3 - onako
  - 4 - verovatno
  - 5 - vrlo verovatno
- Godina na koledzu
- Grčka hrana - koliko je verovatno da bi je jeli
- Koliko se zdravo osećaju od 1 do 10
- Šta je, po njihovim rečima, zdrav obrok
- Šta je, po njihovim rečima, idealna ishrana
- Idealna ishrana - kodirano
- Zarada - koliko zarađuju godišnje
  - 1 - manje od 15 000 dolara
  - 2 - 15 001 - 30 000 dolara
  - 3 - 30 001 - 50 000 dolara
  - 4 - 50 001 - 70 000 dolara
  - 5 - 70 001 - 100 000 dolara
  - 6 - više od 100 000 dolara
- Indijska hrana - koliko je verovatno da bi je jeli
- Italijanska hrana - koliko je verovatno da bi je jeli
- Koliko je život lep od 1 do 10
- Bračni život
  - 1 - slobodan/slobodna
  - 2 - u vezi
  - 3 - živi sa partnerom
  - 4 - udat/oženjen
  - 5 - razveden/a
  - 6 - udovica

- Šta bi poslužili kao večeru prijatelju
- Majčino obrazovanje
- Majčina profesija
- Koliko često proveravaju količinu kalorija hrane koje konzumiraju
- Da li žive ili ne na kampusu
- Koliko puta nedeljno roditelji kuvaju
  - 1 - skoro svaki dan
  - 2 - 2 do 3 puta
  - 3 - 1 do 2 puta
  - 4 - za vreme odmora
  - 5 - nikad
- Plaćanje obroka - koliko bi novca potrošili na obrok
- Persijska hrana - koliko je verovatno da bi je jeli
- Procena sopstvene težine
  - 1 - vitak/vitka
  - 2 - vrlo utreniran/utrenirana
  - 3 - tačno kako treba
  - 4 - pomalo debeo/debela
  - 5 - debeo/debela
  - 6 - ne razmišljam o tome kada mislim o sebi
- Koja od 2 slike ih više asocira sa supom
- Sport - da li se bave sportom
  - 1 - da
  - 2 - ne
  - 99 - bez odgovora
- Tajlandska hrana - koliko je verovatno da bi je jeli
- Koliko tortilja ima kalorija
- Koliko guska ima kalorija
- Sport - kojim sportom se bave
- Povrće - koliko je verovatno da jedu povrće na dnevnoj bazi
- Vitamini - da li uzimaju dodatne vitamine
- Koliko vafli ima kalorija
- Težina (u funtama:  $1\text{kg} = 2.20462\text{ funti}$ )



## 2 Detaljnije upoznavanje sa skupom podataka

Pogledajmo koliko ima ispitanika kog pola:

```
# kolonu Gender prebacujemo u brojevni tip i stampamo dobijene tabele  
# zenski pol je kodiran brojem 1, a muski brojem 2
```

```
data.Gender = data.Gender.astype(int)  
print(data[data['Gender'] == 1])  
print(data[data['Gender'] == 2])
```

Nakon prvog ispisa dobijamo tabelu koja je sačinjena samo od ispitanika koji su ženskog pola i prvih nekoliko redova izgleda ovako:

	GPA	Gender	...	waffle_calories	weight
1	3.654	1	...	900	155
2	3.3	1	..	900	I'm not answering this.
3	3.2	1	...	1315	Not sure, 240
4	3.5	1	...	760	190
5	2.25	1	...	1315	190
7	3.3	1	...	1315	137
8	3.3	1	...	760	180

Nakon drugog ispisa dobijamo tabelu koja je sačinjena samo od ispitanika koji su muškog pola i prvih nekoliko redova izgleda ovako:

	GPA	Gender	breakfast	...	vitamins	waffle_calories	weight
0	2.4	2	1	...	1	1315	187
6	3.8	2	1	...	1	1315	180
12	3.4	2	1	...	2	575	264
14	3.1	2	1	...	1	900	185
15	NaN	2	2	...	2	1315	180
17	3.6	2	1	...	2	900	170

Ukupno ima 76 ispitanika koji su ženskog pola, 46 koji su muškog.

Već iz ovako jednostavnog baratanja podacima možemo zaključiti nešto:

na osnovu podataka koje imamo - muškarci nemaju problem da kažu koliko su teški, što se može videti iz toga da postoje podaci za skoro sve muške osobe kada je kolona weight u pitanju. Kada se pogleda kolona weight za ženske ispitanike vidi se da postoje neke od njih koje imaju problem da kažu svoju težinu.

Podatke ćemo grupisati po polu i onda izračunati mean za svaki od atributa i možda zaključiti još nešto.

```
# pravi se novi skup gde su podaci grupisani na osnovu pola ispitanika  
# u nasem slucaju ce biti grupisani u 2 reda
```

```

groupby_gender = data.groupby('Gender')

# kako nije moguće videti ceo skup,
# skup se mora odstampati na poseban način

with pd.option_context('display.max_rows', \
    None, 'display.max_columns', 100):
    print(groupby_gender.mean())

```

Izdvojene su neke, nama interesantne, kolone:

Gender	comfort_food_reasons_coded	cook
1	2.419355	2.527027
2	3.090909	3.187500

Gender	eating_out	exercise	healthy_feeling
1	2.552632	1.661538	5.328947
2	2.571429	1.489362	5.653061

Gender	self_perception_weight	veggies_day
1	3.320000	4.131579
2	2.816327	3.816327

Na osnovu srednje vrednosti svake kolone posebno za muškarce i posebno za žene vidimo da, u proseku, žene svoju omiljenu hranu konzumiraju iz dosade i depresivnog raspoloženja, dok muškarci najviše zbog depresivnog raspoloženja.

Sa druge strane, potpuno očekivano, žene kuvaju što češće mogu. Za muškarce se može ništa zaključiti na osnovu srednje vrednosti ovog atributa koja je 3.18 i tačno između odgovora da kuvaju kad god mogu i da kuvaju jako retko.

Pripadnici oba pola se prosečno prejedaju i vežbaju jednako. Na pitanje koliko se zdravo osećaju su odgovorili otprilike približno.

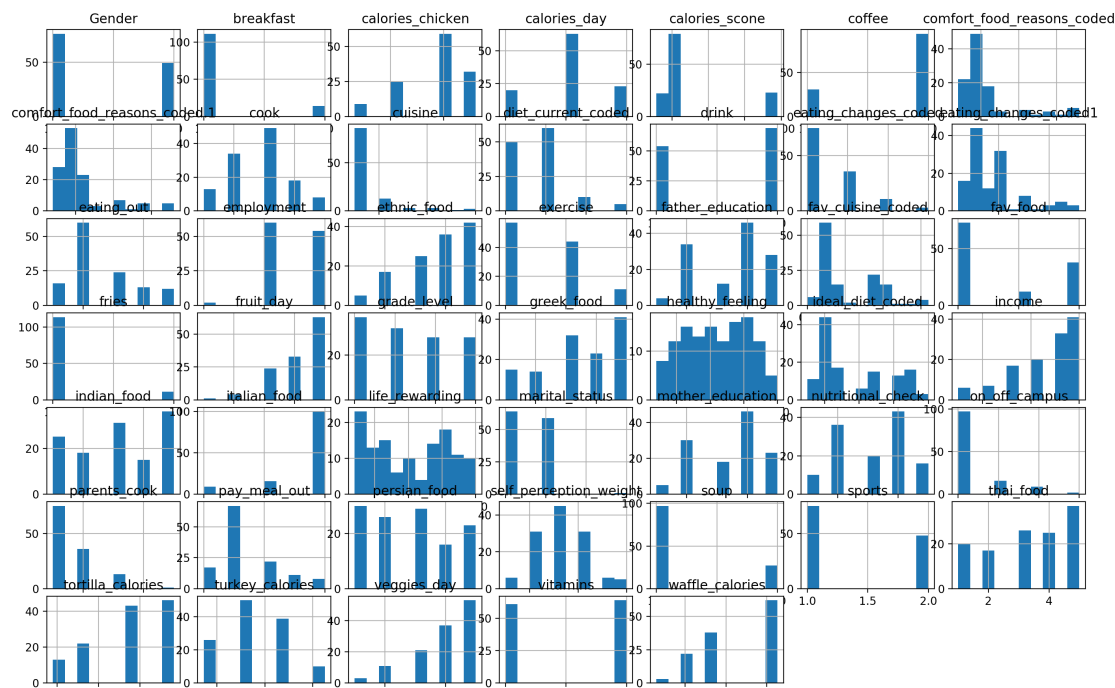
Na to kako procenjuju svoju težinu, većina muškaraca je odgovorila da se oseća utrenirano, dok su žene ponovo malo nesigurnije - odgovori su između "otprilike kako treba" i "malo ugojeno". U isto vreme, šansa da žene jedu povrće dnevno je za nijansu veća od onih koju procenjujemo za muške ispitanike.

```

# korisno je i pogledati skup
# nad citavim skupom, tacnije, za svaku kolonu
# napravit se histogram

data.hist()
plt.show()

```



Na osnovu histograma se jednostavno zaključuje da ima mnogo više žena, na primer. Za pitanja - atribut, koja su oblika "Od ponuđene dve slike izabrati...", lako se može dobiti odgovor šta su ljudi odgovarali u proseku.

Tako, velika većina njih je na pitanje šta za njih predstavlja doručak odgovorila sa "žitarice", velika većina njih misli da je umereno važno koliko se kalorija unosi dnevno. Veliki broj ispitanika svoju omiljenu hranu jede iz dosade, zbog stresa i depresije, dok zanemarljiv broj njih jede zbog gladi (što bi trebalo da je glavni razlog zbog koga konzumiramo hranu). Takođe, većina njih se bavi fizičkom aktivnošću. Na osnovu nekoliko histograma, može se videti da je veliki broj ispitanika ljubitelj italijanske hrane: paste, testenina, pice, dok jako veliki broj njih ne bi okusilo indijsku hranu.

### 3 Primer klasifikacije

Klasifikovaćemo podatke 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:".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, izdvojimo sledeće attribute: `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 niske u nekoj od ove četiri kolone. Takođe, vrednosti u koloni weight ćemo transformisati. Preslikaćemo ih u skup {0, 1, 2}. Dakle, želimo da klasifikujemo ispitanike po težini, u 3 grupe, u zavisnosti od toga koliko se često prejedaju, koliko često kuvaju i zašto jedu hranu za utehu. U prvu grupu spadaju oni koji su teški do 150 funti (oko 70kg), u drugu oni koji imaju između 70kg i 90 kg i u treću grupu oni sa preko 90kg.

Prvo se brišu nedostajuće vrednosti.

```
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("\nTrening_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("\nMatrica_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]]
```

Kao što se može videti iz matrica konfuzije za test i trening skup, podaci su dosta dobro klasifikovani - broj tačno klasifikovanih je dosta veliki, za razliku od onih koji to nisu.

## 4 Vizuelizacija podataka

Atributi koje posmatramo:

- self\_perception\_weight  
Procena sopstvene težine
- fruit\_day  
Koliko je verovatno da bi pojeli voće u toku dana
- pay\_meal\_out  
Koliko bi platili za jedan obrok?
- weight  
Težina

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from mpl_toolkits.mplot3d import Axes3D

# read data
df = pd.read_csv('./food-choices/food_coded.csv')

# choose attributes
target_attribute = 'weight'
attribute_1 = 'self_perception_weight'
attribute_2 = 'fruit_day'
attribute_3 = 'pay_meal_out'

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

# data preprocessing

# remove NaN
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[self_perception_weight].astype(int)
df[attribute_2] = df[fruit_day].astype(int)
df[attribute_3] = df[pay_meal_out].astype(int)
df[target_attribute] = df[weight].astype(int)

# transform target attribute
changes = {}
```



```

weight = df[target_attribute].unique()
for w in weight:
    if int(w) <=128:
        changes[w] = 0
    elif int(w) <= 155:
        changes[w] = 1
    elif int(w) <= 180:
        changes[w] = 2
    else:
        changes[w] = 3

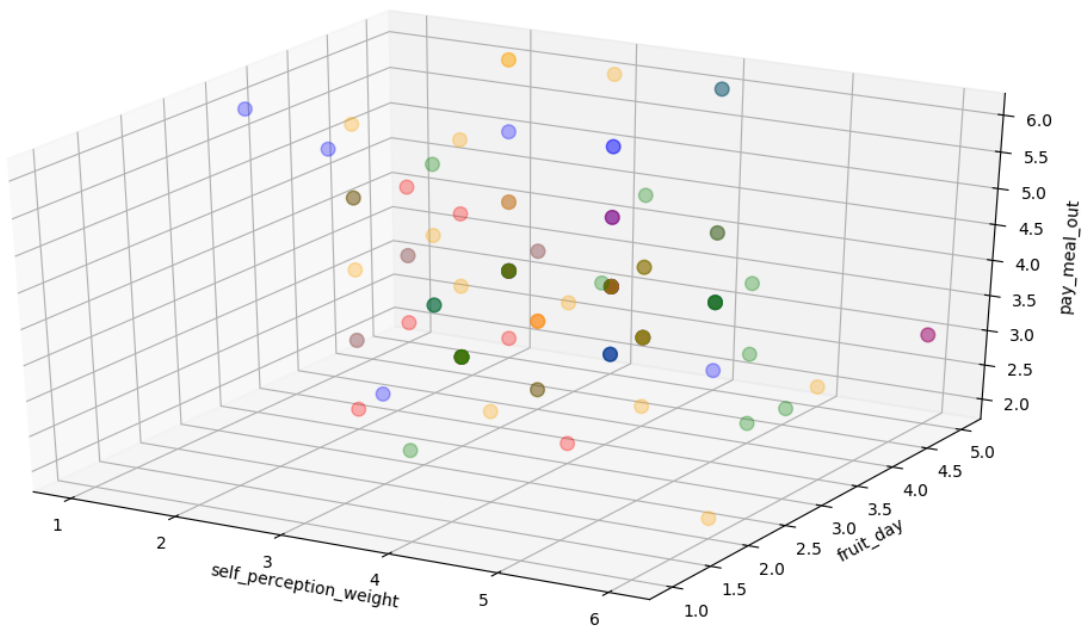
df[target_attribute] = df[target_attribute].replace(changes)
weight = df[target_attribute].unique()

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')

colors = ['green', 'blue', 'orange', 'red']
for (v, color) in zip(weight, colors):
    subsamples = df.loc[df[target_attribute] == v]
    ax.scatter(subsamples[attribute_1],
               subsamples[attribute_2],
               subsamples[attribute_3],
               color=color, s=70, alpha=0.3)

ax.set_xlabel('self_perception_weight')
ax.set_ylabel('fruit_day')
ax.set_zlabel('pay_meal_out')
plt.show()

```



## 4.1 PCA

Korišćenjem PCA algoritma ćemo redukovati skup na manji, zadržavajući većinu informacija iz inicijalnog skupa podataka.

```
from sklearn.decomposition import PCA
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import StandardScaler

def isFloat(value):
    try:
        float(value)
        return True
    except ValueError:
        return False

df = pd.read_csv('./food-choices/food_coded.csv')

# odabir atributa

target_attribute = 'weight'
attributes = [ 'exercise', 'Gender', 'eating_out',
               'GPA', 'employment', 'breakfast',
               'calories_chicken', 'calories_day', 'coffee',
               'diet_current_coded', 'drink', 'cook' ]

df = df[[ attributes[0], attributes[1], attributes[2],
```

```

        attributes[3], attributes[4], attributes[5],
        attributes[6], attributes[7], attributes[8],
        attributes[9], attributes[10], attributes[11],
        target_attribute]]

# obrada podataka
# brisanje redova sa nedostajucim vrednostima,
# konvertovanje podataka u odgovarajuce
# kako bi se moglo raditi sa njima

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

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

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

# podaci su tipa float, a potrebno je
# da budu tipa int, pa se u narednim
# koracima vrši konverzija

df[attributes[0]] = df.exercise.astype(int)
df[attributes[1]] = df.Gender.astype(int)
df[attributes[2]] = df.eating_out.astype(int)
df[attributes[3]] = df.GPA.astype(float)
df[attributes[4]] = df.employment.astype(int)
df[attributes[5]] = df.breakfast.astype(int)
df[attributes[6]] = df.calories_chicken.astype(int)
df[attributes[7]] = df.calories_day.astype(int)
df[attributes[8]] = df.coffee.astype(int)
df[attributes[9]] = df.diet_current_coded.astype(int)
df[attributes[10]] = df.drink.astype(int)
df[attributes[11]] = df.cook.astype(int)
df[target_attribute] = df.weight.astype(int)

# transformisanje ciljnog atributa
# preslikacemo sve vrednosti weight atributa
# u skup {0, 1, 2, 3}
# ispitanici:
# sa manje od 58.5kg spadaju u grupu 0
# izmedju 58.5 i 70kg spadaju u grupu 1
# izmedju 70 i 81.6kg spadaju u grupu 2
# sa preko 81.6kg spadaju u grupu 3

changes = {}
weight = df[target_attribute].unique()

```

```

for w in weight:
    if int(w) <=128:
        changes[w] = 0
    elif int(w) <= 155:
        changes[w] = 1
    elif int(w) <= 180:
        changes[w] = 2
    else:
        changes[w] = 3

df[target_attribute] = df[target_attribute].replace(changes)
weight = df[target_attribute].unique()

features = attributes

# extract features
x = df.loc[:, features].values
# extract target attribute
y = df.loc[:, [target_attribute]].values

x = StandardScaler().fit_transform(x)

pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principalDf = pd.DataFrame(data = principalComponents
                           , columns=['principal_component_1', 'principal_component_2'])

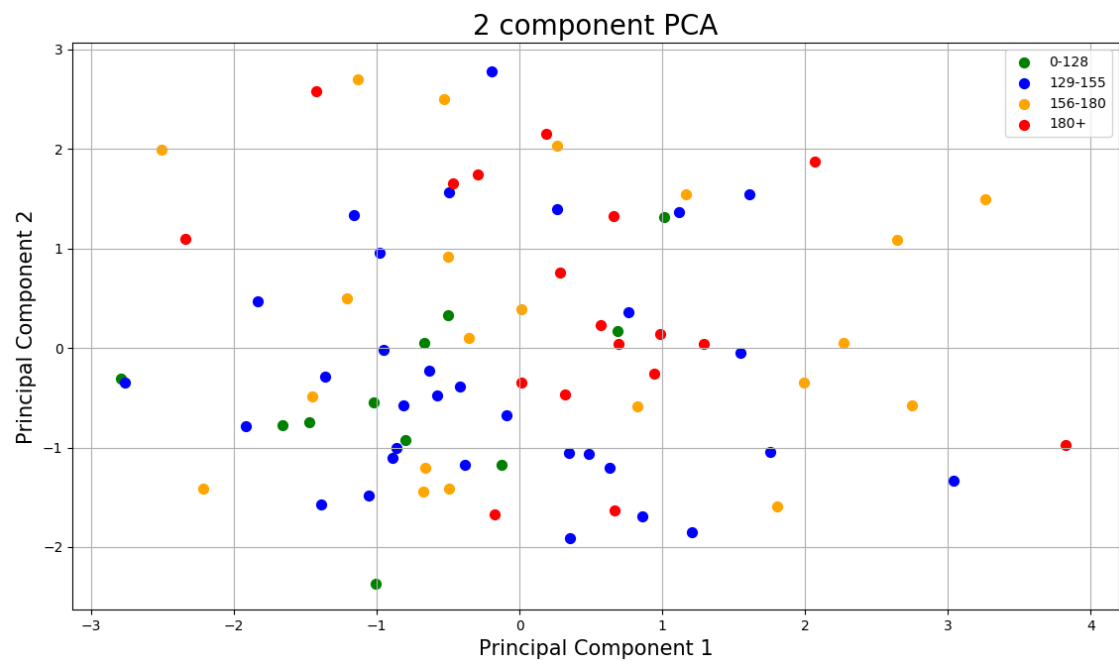
finalDf = pd.concat([principalDf, df[[target_attribute]]], axis = 1)

fig = plt.figure(figsize = (8,8))
ax = fig.add_subplot(1,1,1)
ax.set_xlabel('Principal_Component_1', fontsize = 15)
ax.set_ylabel('Principal_Component_2', fontsize = 15)
ax.set_title('2_component_PCA', fontsize = 20)

targets = [0,1,2,3]
colors = ['green', 'blue', 'orange', 'red']
for target, color in zip(targets, colors):
    indicesToKeep = finalDf[target_attribute] == target
    ax.scatter(finalDf.loc[indicesToKeep, 'principal_component_1']
               , finalDf.loc[indicesToKeep, 'principal_component_2']
               , c=color
               , s=50)

ax.legend(['0-128', '129-155', '156-180', '180+'])
ax.grid()
plt.show()

```



Na osnovu grafika zaključujemo da najveći broj ispitanika ima između 58.5kg i 70kg.