1 Rad sa velikim skupom podataka

Koristimo dataset preporuka o poljoprivrednim vrstama (preporuke_d.csv). Svaki red ima 8 kolona. Prvih 7 kolona su mjerenja koja su dobijena analizom lokaliteta (zemljište, vazduh i klima) a posljednja (osma) kolona predstavlja vrstu koja je zasađena na analiziranom lokalitetu.

1.1 Postavljanje okruženja

Prije svega učitajmo biblioteke koje će nam biti potrebne. Više informacija o bibliotekama na [1],[2].

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Učitavamo dataset:

```
df = pd.read_csv('preporuke_d.csv')
```

Ignorišemo redove sa nedostajućim podacima na sledeći način:
Primjetimo da su nedostajuće kolone prazne. Da bi iskoristili funckiju pandas.DataFrame.dropna() koja ignoriše redove u kojima se nalaze NA vrijednosti, moramo prvo zamijeniti prazne stringove sa NaN. Argument inplace = True omogućava da se DataFrame objekat izmijeni i da se izmjene sačuvaju u istom objektu.

```
nan_value = float('NaN')
df.replace("", nan_value, inplace=True)
df.dropna(inplace=True)
```

1.2 Analiza dataset-a

Izračunavanjem statistika (npr. srednje vrijednosti, opsezi vrijednosti) i grafičkim predstavljanjem podataka analizirati dataset. Cilj je odgovoriti na pitanja :

1. Koje vrste uspijevaju i na lokalitetima sa malim brojem padavina?

2. Koje vrste uspijevaju na lokalitetima sa visokim nivoom azota?

3. Koja vrsta uspijeva na najnižim a koja na najvišim temperaturama?

4. Kojim vrstama je neophodan visok nivo fosfora a kojima je nizak nivo fosfora dovoljan?

5. Analizirati nivo vlažnosti na lokalitetima na kojima uspijevaju pojedinačne vrste?

6. Ako znamo da su prema pH vrijednosti zemljišta poređana u sledeće kategorije, koje vrste dominantno uspijevaju u kojim kategorijama?

   pH < 4.5 – veoma kisela zemljišta
   pH od 4.5 do 5.5 – kisela zemljišta
   pH od 5.6 do 6.7 – umereno kisela zemljišta
   pH od 6.8 do 7.2 – neutralna zemljišta
   pH > 7.2 – alkalna (bazična ili bazna) zemljišta

Na pitanja 1 i 2 odgovoraćemo na sličan način.

Potrebno je naći lokalitete sa malim brojem padavina i naći koje vrste uspijevaju na tim lokalitetima. U obzir ćemo uzeti količine padavina koje se nalaze u prvom kvartilu.

```
granica = data_set['padavine'].quantile(.25)
vrste_padavine = data_set.loc[data_set['padavine']<=granica]
```

Nakon što smo iz dataset-a izdvojili redove koji zadovoljavaju taj uslov, potrebno je naći koje su to vrste koje uspijevaju na takvim lokalitetima. Dijagram će izgledati tako da na x osi budu nazivi vrsta koje uspijevaju na ovakvim lokalitetima, a na y osi procenat takvih lokaliteta u odnosu na ukupan broj lokaliteta na kojima vrsta uspijeva.

Izdvajamo redove za vrste koje se nalaze u DataFrame-u vrste_padavina i računamo ukupan broj lokaliteta na kojima vrste uspijevaju.
Funckija numpy.intersect1d nalazi presjek dva array like elementa. Funkcija pandas.DataFrame.groupby() grupise podatke po koloni koja je zadata argumentom, a .count() broji elemente u svakoj grupi.

```
val = np.intersect1d(data_set.vrsta,vrste_padavine.vrsta)
temp = data_set[data_set.vrsta.isin(val)]
ukupan_broj_lokaliteta = temp.groupby('vrsta').count()
```

Računamo potreban procenat.

```
broj_lokaliteta = (vrste_padavine.groupby('vrsta').count())/
                                    ukupan_broj_lokaliteta*100
```

Na primjer, za vrstu lentil vidimo da ona uspijeva samo(100%) na lokalitetima sa malim brojem padavina.
Koršićenjem biblioteke matplotlib generišemo dijagram (Figure 1). Funckija .plot() generisaće line dijagram.

```
plt.figure(figsize=[10,6])
plt.title('Lokaliteti sa kolicinom padavina <= %i' %granica)
plt.plot(broj_lokaliteta.padavine,color='C0')

plt.subplots_adjust(bottom=0.234,left=0.134)
plt.grid(True)

plt.xticks(rotation=90)
plt.xlabel('vrste')
plt.ylabel('procenat lokaliteta (%)')

plt.savefig('padavine')
plt.show()
```

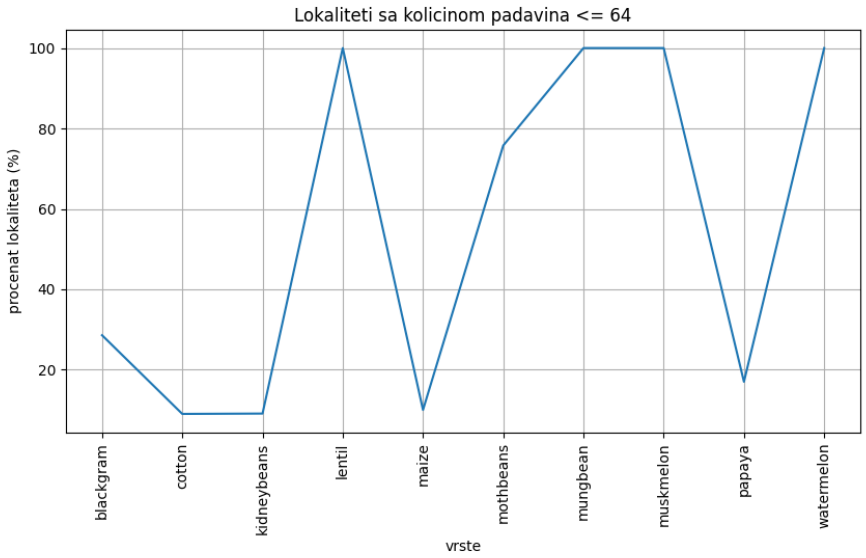

Figure 1: Pitanje 1

Analogno ovom načinu razmišljanja generišemo dijagram za pitanje 2. Tražimo lokalitete sa visokim nivoom azota.

```
granica = data_set.azot.quantile(.75)
vrste_azot = data_set.loc[data_set['azot']<=granica]

val = np.intersect1d(data_set.vrsta,vrste_azot.vrsta)
temp = data_set[data_set.vrsta.isin(val)]
ukupan_broj_lokaliteta = temp.groupby('vrsta').count()
broj_lokaliteta = (vrste_azot.groupby('vrsta').count())
                            /ukupan_broj_lokaliteta*100

plt.figure(figsize=[10,6])
plt.title('Lokaliteti sa kolicinom azota >= %i' %granica)
plt.plot(broj_lokaliteta,color='C0',linewidth=1.0)
```

Dodatno, označimo na dijagramu one vrste koje uspijevaju isključivo na lokalitetima sa visokim nivoom azota pomoću scatter plot-a.

```
x = broj_lokaliteta.loc[broj_lokaliteta['azot']==100]
```

```
x = list(x.index)
y = np.ones(len(x))*100
plt.scatter(x,y,marker='o',color='r')
```

y je niz koji sadrži onoliko elemenata koliko i niz x, pri čemu su svi jednaki 100.

```
plt.subplots_adjust(bottom=0.234,left=0.134)
plt.grid(True)

plt.xticks(rotation=90)
plt.xlabel('vrste')
plt.ylabel('procenat lokaliteta (%)')

plt.savefig('azot')
plt.show()
```

Dobijamo dijagram (Figure 2):

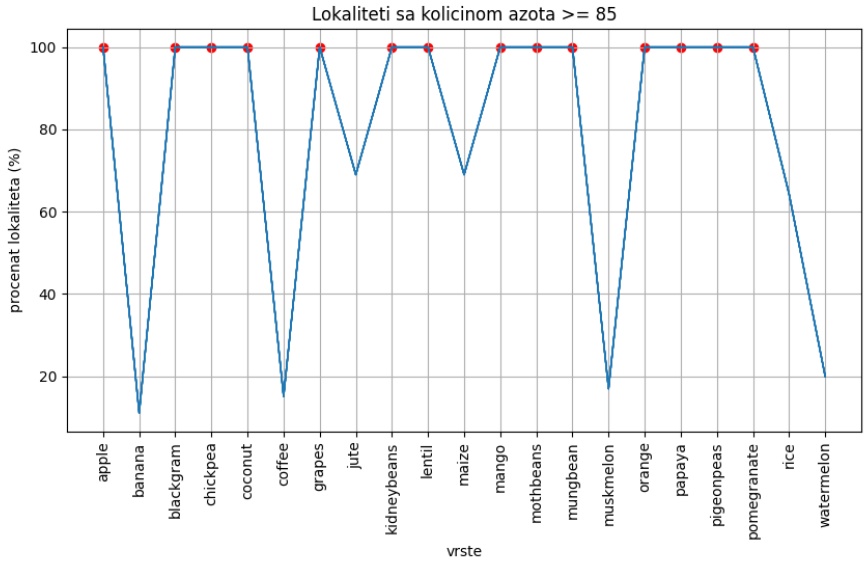

Figure 2: Pitanje 2

Na pitanja 3 i 4 odgovorićemo na sličan način. Dijagram generišemo na osnovu prosjčenih temeperatura na lokalitetima na kojima uspijevaju pojedinačne vrste.

Na dijagramu ćemo označiti i koje su to granice koje odvajaju najveće i najmanje temperature.

```
min_temp = data_set['temp'].quantile(.25)
max_temp = data_set['temp'].quantile(.75)

temperatura_vrsta = data_set[['temp','vrsta']].groupby('vrsta').mean()

plt.figure(figsize=[10,6])
plt.plot(temperatura_vrsta,color='C0')
line1 = plt.axhline(y=min_temp, color='g', linestyle='--', linewidth=1)
line2 = plt.axhline(y=max_temp, color='r', linestyle='--', linewidth=1)

plt.subplots_adjust(bottom=0.293,right=0.915)
plt.grid(True)
plt.title('Temperatura i vrsta')
plt.xticks(rotation=90)
plt.xlabel('vrsta')
plt.ylabel('prosjecna temperatura')

plt.legend([line1,line2],['Q1','Q3'])
plt.savefig('temperatura')
plt.show()
```

Dobijamo dijagram (Figure 3):

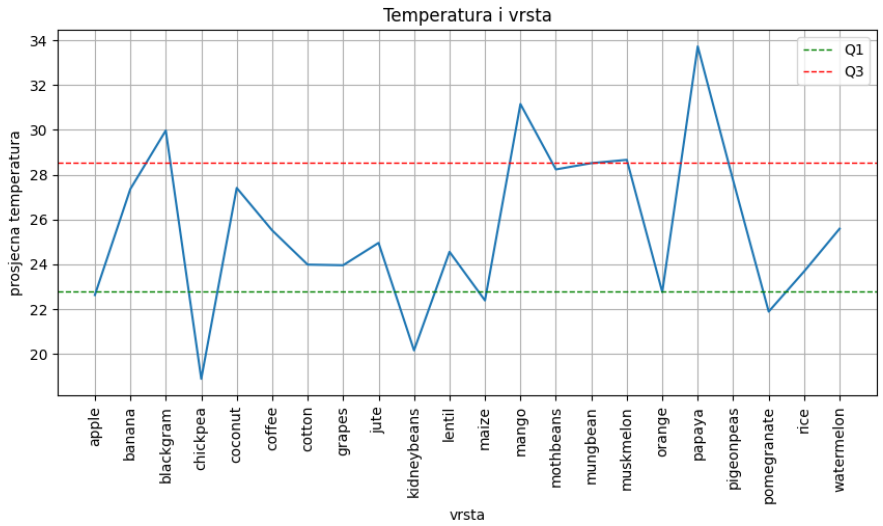

Figure 3: Pitanje 3

Analogno, dobijamo odgovor na pitanje 4 i dijagram(Figure 4) samo radimo sa kolonom fosfor.

```
min_fosfor = data_set['fosfor'].quantile(.25)
max_fosfor = data_set['fosfor'].quantile(.75)

fosfor_vrsta = data_set[['fosfor','vrsta']].groupby('vrsta').mean()

plt.figure(figsize=[10,6])
plt.plot(fosfor_vrsta,color='C0')
line1 = plt.axhline(y=min_fosfor, color='g', linestyle='--', linewidth=1)
line2 = plt.axhline(y=max_fosfor, color='r', linestyle='--', linewidth=1)

plt.subplots_adjust(bottom=0.22)
plt.grid(True)
plt.title('Kolicina fosfora i vrste')
plt.xticks(rotation=90)
plt.xlabel('vrste')
plt.ylabel('kolicina fosfora')
```

```
plt.legend([line1,line2],['Q1','Q3'])
plt.savefig('fosfor.png')
plt.show()
```

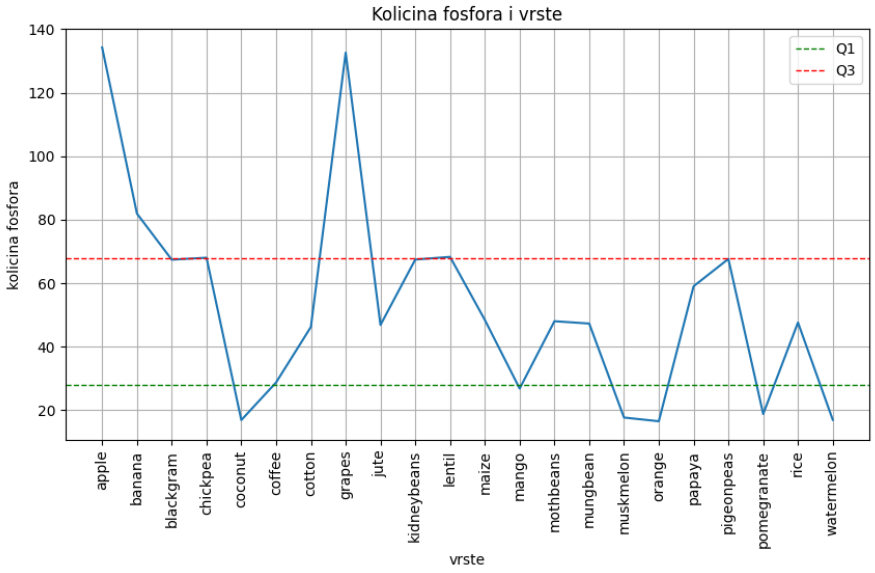

Figure 4: Pitanje 4

Za pitanje 5 generišemo boxplot dijagram (Figure 5) - prikazana je distribucija podataka(vlažnost lokaliteta za svaku vrstu) na osnovu 5 brojeva - minimum,prvi kvartil,medijan,treći kvartil i maksimum. Dodatno prikazimo i projsečnu vrijednost za svaku vrstu.

```
vlaznost_vrsta = data_set[['vlaznost','vrsta']].groupby('vrsta')

vlaznost_vrsta.boxplot(figsize=(10,6),subplots=False)

vlaznost_vrsta = vlaznost_vrsta.mean()

plt.plot(range(1,len(vlaznost_vrsta)+1),vlaznost_vrsta.vlaznost,
                              color='r',linewidth=0.8,label='mean')
```

Za generisanje boxplot dijagrama koristimo funckiju pandas.DataFrame.boxplot() za kolonu koja sadrži nivo vlažnosti. Pošto koristimo i funkciju matplotlib.pyplot.plot za prosječne vrijednosti, moramo pomjeriti dijagram duž x ose.

```
plt.subplots_adjust(bottom=0.293,right=0.915)
plt.grid(True)
plt.title('Vlaznost i vrsta')

xticklabels = vlaznost_vrsta.index
plt.xticks(range(1,len(xticklabels)+1),labels=xticklabels, rotation=90)

plt.xlabel('vrsta')
plt.ylabel('vlaznost')

plt.legend()
plt.savefig('vlaznost')
plt.show()
```

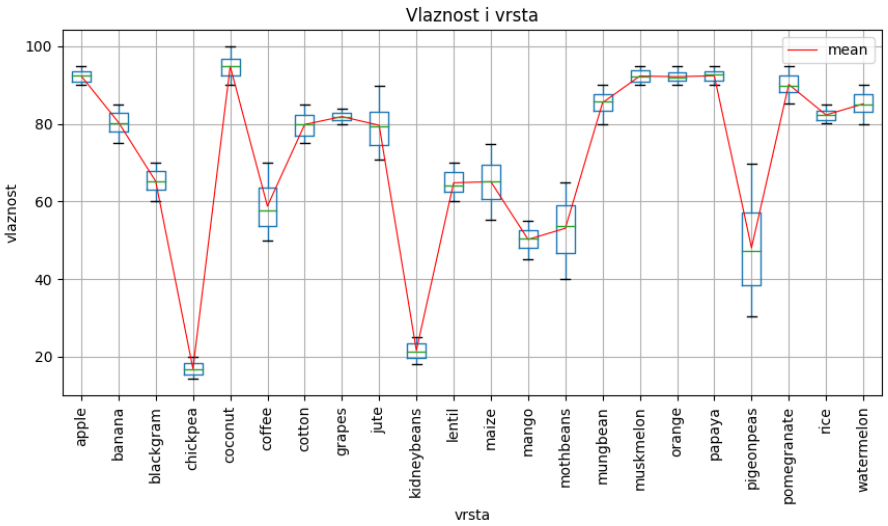

Figure 5: Pitanje 5

Da bi odgovorili na pitanje 6 prije svega definisaćemo jednostavnu funckiju koja klasifikuje lokalitet u jednu od 5 grupa. Fuckija za argument uzima pH vrijednost zemljišta.

```
def ph_fja(ph_vr):
    if(ph_vr<4.5): return 'veoma kisela'
    elif(ph_vr<=5.5): return 'kisela'
    elif(ph_vr<=6.7): return 'umjereno kisela'
    elif(ph_vr<=7.2): return 'neutralna'
    else: return 'alkalna'
```

Ovim dijelom koda izdvajamo kolone vrsta i ph_vrijednost, a onda dodajemo novu kolonu naziv_grupe koja nam govori kojoj pH grupi lokalitet pripada.
Koristimo .copy() da izbjegnemo SettingWithCopyWarning izazvano ulančanim dodjeljivanjem.
Argument axis = 1 je indikator da se aplikacija funckije vrši na nivou reda, a ne kolone.

```
ph_grupa = data_set[['vrsta','ph_vrijednost']].copy()
ph_grupa.loc[:,'naziv_grupe'] = ph_grupa.apply
        (lambda row:ph_fja(row.ph_vrijednost),axis=1)
```

Grupisanje vrsimo po dvije kolone - vrsta i naziv_grupe, a zatim računamo procentualno koje vrste uspijevaju u kojoj ph grupi.

```
grouped = ph_grupa.groupby(['naziv_grupe','vrsta'])
            .count().rename(columns={'ph_vrijednost':'broj'})
result = grouped/grouped.groupby(level=0).sum()*100
```

Ova lista boja (22 boje) generisana je pomoću [3]. Za do 20 boja mogli smo iskoristiti neku od ugrađenih color map-a. Na primjer: colors = plt.cm.tab20c(np.linspace(0,1,22))

```
colors = ["#d58840",
        "#5f36b7",
        "#79d645",
        "#c44bca",
        "#d0cd3c",
        "#6b67c2",
        "#63d689",
        "#c74381",
        "#4c7e34",
        "#cc91cf",
        "#c2d07c",
        "#482752",
        "#79ceb6",
        "#d6483b",
        "#95c3d9",
        "#7b312e",
        "#6380ac",
        "#887035",
        "#bc7a83",
        "#39362a",
        "#d4bda3",
        "#527664"]
```

Koristimo pandas.DataFrame.unstack() gdje kao rezultat dobijamo DataFrame jer korisitmo MultiIndex. Za dobijeni DataFrame korsitimo pandas.DataFrame.plot.bar() da bi dobili bar chart, a argument stacked = True nam daje stacked bar chart.

```
result.unstack().plot.bar(stacked = True,
            figsize = (10,6),color=colors)
plt.title('Lokaliteti i pH grupe')
plt.subplots_adjust(bottom=0.228)
plt.xticks(rotation=30)
plt.xlabel('grupe')
plt.ylabel('procenat (%)')
```

U legendu dodajemo nazive vrste koji odgovaraju bojama. Pošto je grouped DataFrameGroupBy objekat koristimo .index.get_level_values().

```
plt.legend(labels=grouped.index.get_level_values(1).unique(),
                                                prop={'size': 8})
plt.savefig('ph.png')
plt.show()
```

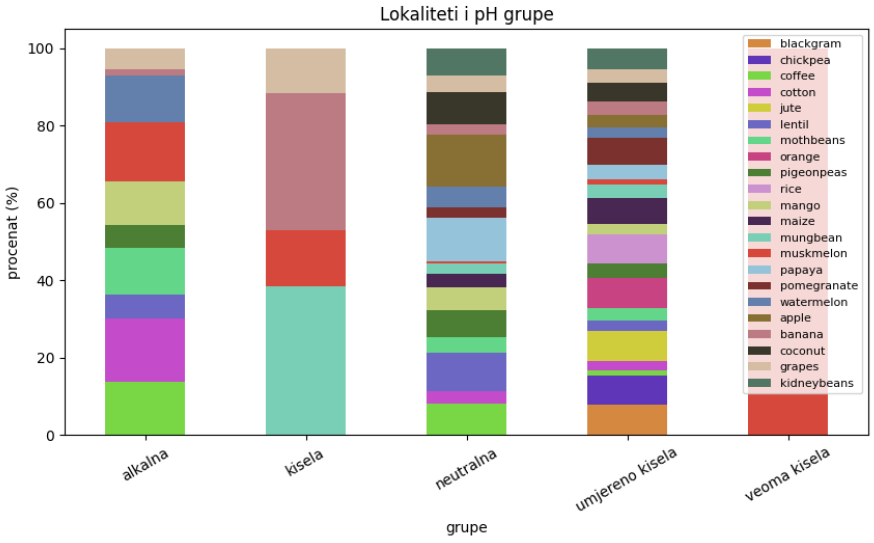

Figure 6: Pitanje 6

Literatura

[1] https://pandas.pydata.org/docs/index.html

[2] https://matplotlib.org/stable/

[3] https://medialab.github.io/iwanthue/