Języki programowania w analizie danych

Zadanie 1. na zbiorach Iris oraz Quakes

Maciej Majchrowski 234088

1. Opis zbiorów danych

Iris:

„Title: Iris Plants Database
Updated Sept 21 by C.Blake - Added discrepency information

Number of Attributes: 4 numeric, predictive attributes and the class

Attribute Information:
1. sepal length in cm
2. sepal width in cm
3. petal length in cm
4. petal width in cm
5. class:
   -- Iris Setosa
   -- Iris Versicolour
   -- Iris Virginica”

Quakes:

„Description

The data set give the locations of 1000 seismic events of MB > 4.0. The events occurred in a cube near Fiji since 1964.

A data frame with 1000 observations on 5 variables.

[,1]	lat	numeric	Latitude of event
[,2]	long	numeric	Longitude
[,3]	depth	numeric	Depth (km)
[,4]	mag	numeric	Richter Magnitude
[,5]	stations	numeric	Number of stations reporting”

2. Opis działania programu

1. Wybranie zestawu do obliczeń oraz konfiguracja

```
{
    "chosenDataSet":"iris",
    "iris":{
        "dataSourceUrl": "./data/iris.csv",
        "testDataSize": 0.2,
        "quantityFactors": false,
        "numberedFirstColumn": 0,
        "numberedLastColumn": 3,
        "classColumn": "class"
    },
    "quakes":{
        "dataSourceUrl": "./data/quakes.csv",
        "testDataSize": 0.2,
        "quantityFactors": false,
        "numberedFirstColumn": 1,
        "numberedLastColumn": 5,
        "classColumn": "Position"
    }
}
```

Plik konfiguracyjny programu configuration.json.
Wybór zestawu danych polega na dopasowaniu pola chosenDataSet

2. Klasa Data - klasa otrzymująca zestaw danych, dzieląca go na cechy jakościowe oraz ilościowe, zwracająca podstawowe informacje statystyczne

```
54      def getMedians(self):
55          dict = {}
56          for quantitativeLabel in self.QuantitiveColumns:
57              dict[quantitativeLabel] = statistics.median(self.dataset[quantitativeLabel])
58          return dict
59
60      def getMaximum(self):
61          dict = {}
62          for quantitativeLabel in self.QuantitiveColumns:
63              dict[quantitativeLabel] = max(self.dataset[quantitativeLabel])
64          return dict
65
66      def getMinimum(self):
67          dict = {}
68          for quantitativeLabel in self.QuantitiveColumns:
69              dict[quantitativeLabel] = min(self.dataset[quantitativeLabel])
70          return dict
71
72      def getDominant(self):
73          if self.hasQualityFeatures:
74              dict = {}
75              for label in self.QualityColumns:
76                  dict[label] = Counter(self.dataset[label]).most_common(1)
77              return dict
78          else:
79              return "No quality columns"
```

Metody klasy Data zwracające informacje statystyczne, na podstawie cech jakościowych bądź ilościowych

3. Klasa App - klasa z głównym kodem programu.

Zadanie: *Dla poszczególnych atrybutów wyznaczyć medianę, minimum i maximum dla cech ilościowych i dominantę dla cech jakościowych.*

Dla zestawu Iris zostały obliczone: mediana, minimum i maximum dla cech ilościowych:

```
12          dataset = readcsv_fromFile(config.dataSourceUrl)
13          CurrentDataSet = Data(dataset, config)
14          print("-----Medians for quantitative features----")
15          print(yaml.dump(CurrentDataSet.getMedians(), default_flow_style=False))
16          print("-----Maximum for quantitative features----")
17          print(yaml.dump(CurrentDataSet.getMaximum(), default_flow_style=False))
18          print("-----Minimum for quantitative features----")
19          print(yaml.dump(CurrentDataSet.getMinimum(), default_flow_style=False))
20          print("-----Dominant for quality features----")
21          print(CurrentDataSet.getDominant())
```

Prezentacja wykorzystania metod klasy Data do wyświetlenia wartości poszczególnych atrybutów

```
['sepal-length' 'sepal-width' 'petal-length' 'petal-width']
-----Medians for quantitative features----
petal-length: 4.35
petal-width: 1.3
sepal-length: 5.8
sepal-width: 3.0

-----Maximum for quantitative features----
petal-length: 6.9
petal-width: 2.5
sepal-length: 7.9
sepal-width: 4.4

-----Minimum for quantitative features----
petal-length: 1.0
petal-width: 0.1
sepal-length: 4.3
sepal-width: 2.0

-----Dominant for qualitative features----
No Qualitative columns
```

Atrybuty dla zbioru Iris - cechy ilościowe

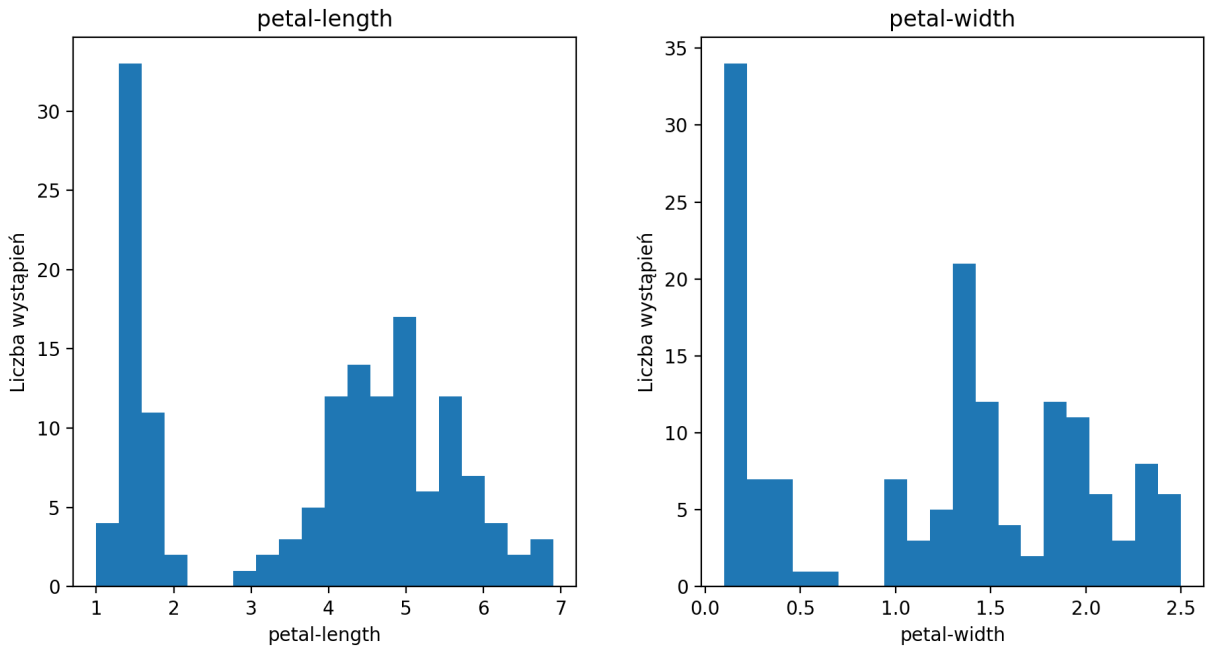
Zadanie: *Narysować histogramy dla dwóch cech ilościowych najbardziej ze sobą skorelowanych.*

```
23        print("-----Correlation matrix for quantitative features----")
24        print(CurrentDataSet.dataset[CurrentDataSet.QuantitiveColumns].corr())
25        corr_matrix = CurrentDataSet.dataset[CurrentDataSet.QuantitiveColumns].corr().abs()
26        bestCorr = (corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.bool))
27                    .stack()
28                    .sort_values(ascending=False))
29        bestCorrStr = str(bestCorr.head(1))
30        a = bestCorrStr.split(" ")
31        firstCorrelationColumn = a[0]
32        secondCorrelationColumn = a[2]
```

Utworzenie macierzy korelacji. Wybranie z macierzy korelacji pary o największej korelacji. Przygotowanie danych do stworzenia histogramu.

```
-----Correlation matrix for quantitative features----
              sepal-length  sepal-width  petal-length  petal-width
sepal-length      1.000000    -0.109369      0.871754     0.817954
sepal-width      -0.109369     1.000000     -0.420516    -0.356544
petal-length      0.871754    -0.420516      1.000000     0.962757
petal-width       0.817954    -0.356544      0.962757     1.000000
```

Macierz korelacji dla zbioru Iris



Histogramy przedstawiające najbardziej skorelowane cechy dla zbioru danych Iris

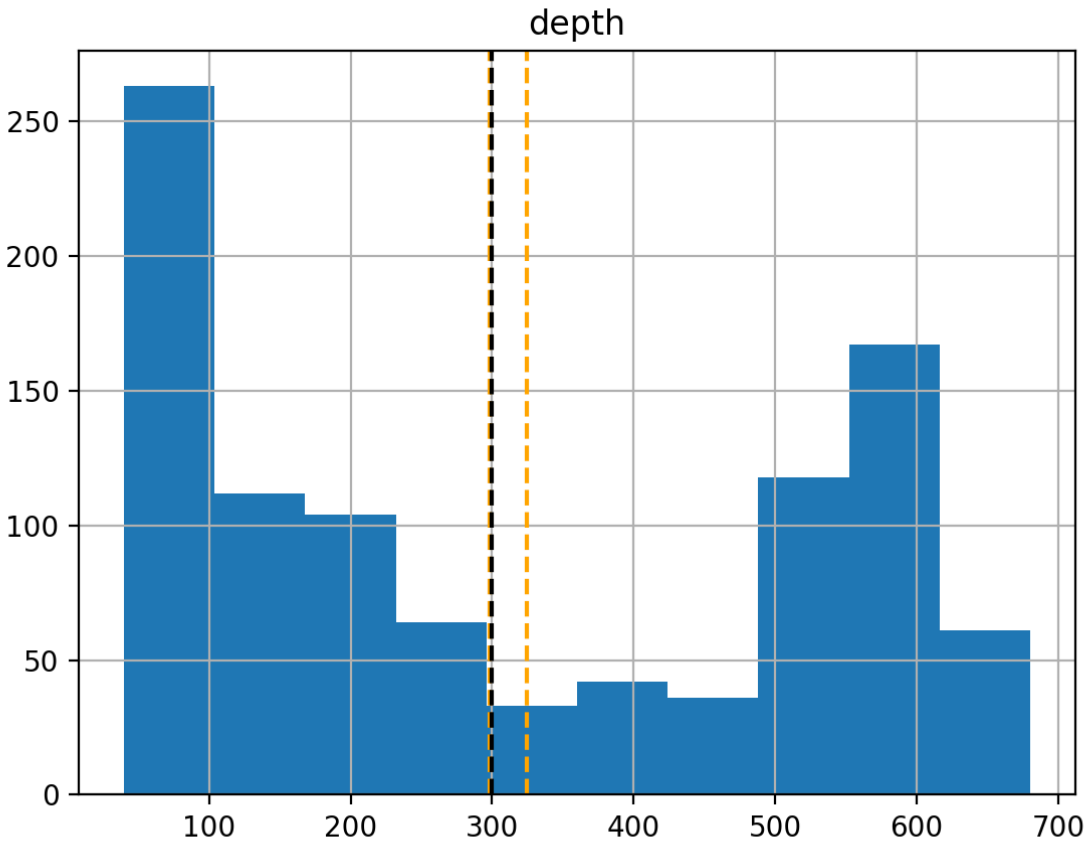
Zadanie: *Dla danych quakes zbadać hipotezę, że średnia głębokość występowania trzęsienia ziemi wynosi 300 metrów. Zwizualizować rozkłady na histogramie. Zaznaczyć na wykresie punkt dotyczący badanej hipotezy.*

```
48          if config.dataSourceUrl == "./data/quakes.csv":
49              hypothesis = 300
50              alpha = 0.05
51              meanD = np.average(CurrentDataSet.dataset["depth"])
52              print("Średnia obliczona ze wszystkich głębokości wystąpienia: " + str(meanD))
53              stdD = np.std(CurrentDataSet.dataset["depth"])
54              print("Odchylenie standardowe głebokości wystąpienia: " + str(stdD))
55              t = stats.t.ppf(1 - alpha/2, CurrentDataSet.dataset["depth"].count()-1)
56              print(f"Wartość krytyczna zbioru quakes w kolumnie depth: {t}")
57              left_crit_boundary = meanD + t*stdD/(sqrt(CurrentDataSet.dataset["depth"].count()))
58              right_crit_boundary = meanD - t*stdD/(sqrt(CurrentDataSet.dataset["depth"].count()))
59              print(left_crit_boundary)
60              print(right_crit_boundary)
61              if hypothesis > right_crit_boundary and hypothesis < left_crit_boundary:
62                  print("Hipoteza potwierdzona")
63              else:
64                  print("hipotera zostaje odrzucona")
```

Przeprowadzenie testowania hipotezy na temat średniej na zbiorze Quakes. Założona hipoteza to: *średnia głębokość występowania trzęsienia ziemi wynosi 300 metrów.*

```
Średnia obliczona ze wszystkich głębokości wystąpienia: 311.371
Odchylenie standardowe głebokości wystąpienia: 215.42770332294776
Wartość krytyczna zbioru quakes w kolumnie depth: 1.9623414611334487
324.73929840820705
298.0027015917929
Hipoteza potwierdzona
```

Obliczenia dążące do potwierdzenia hipotezy. Hipoteza została potwierdzona, ponieważ wynik mieści się w przedziale średniej.



Histogram przedstawiający dane dot. głębokości występowania trzęsień ziemi. Czarna linia - 300 km jako średnia występowania trzęsień ziemi. Pomarańczowe linie - granice testowania hipotezy na temat średniej.

3. Uruchomienie programu

Opisane w pliku README.md załączonym do plików programu.