Problem Set 01: Data Exploration

Johannes C. B. Dietschreit, Sascha Mausenberger

Due: 14.10.2024

To solve the questions in this PSet, download the file `solubility_dataset.csv`. The problems are meant to be solved using Python functions and libraries. You can do so by either writing short scripts or using Jupyter notebooks. **Report your results in the Quiz on Moodle** to obtain a grade.

1 Introduction

In this PSet you will have the chance to familiarize yourself with libraries used when dealing with data sets. We will present you with some introductory code for some important packages, however, we expect you to look for functions you might need yourself (the documentations of all mayor Python packages we use in this course are very good, further Google and ChatGPT are of help).

1.1 Pandas

Pandas is a powerful and flexible open-source data analysis and manipulation library for Python, widely used for working with structured data. It provides data structures like `DataFrames` and `Series`, which allow for efficient handling, analysis, and manipulation of large datasets. It has extensive functionality for data cleaning, merging, and reshaping. Pandas is essential for data scientists and analysts.

```python
import pandas as pd

# Load a CSV file into a DataFrame
df = pd.read_csv('data.csv')

# Display the first few rows of the DataFrame
print(df.head())
# Get basic statistical summary of the data
print(df.describe())

# Calculate mean, median, and standard deviation for a specific column
column_name = 'column_of_interest'
mean_value = df[column_name].mean()
median_value = df[column_name].median()
std_dev_value = df[column_name].std()

# Identify correlations between columns
print(df.corr())

# Count the occurrences of unique values in a specific column
print(df[column_name].value_counts())
```

1.2 Matplotlib and Seaborn

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python, offering a variety of plotting functions and customization options.

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

# Load a CSV file into a DataFrame
df = pd.read_csv('data.csv')

column_name = 'column_of_interest'
# Histogram
plt.figure(figsize=(10, 6))
plt.hist(df[column_name], bins=30, edgecolor='k', alpha=0.7)
plt.title(f'Histogram of {column_name}')
plt.xlabel(column_name)
plt.ylabel('Frequency')
plt.show()
# Scatter plot
plt.scatter(df['x_column'], df['y_column'], alpha=0.7)
plt.title('Scatter Plot of x_column vs y_column')
plt.xlabel('x_column')  # Replace with your x column name
plt.ylabel('y_column')  # Replace with your y column name
plt.show()
# Line plot
plt.plot(df['x_column'], df['y_column'], marker='o')
...
```

Seaborn is a statistical data visualization library based on Matplotlib, designed to create informative and attractive visualizations with minimal code.

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv')
# Plotting with Seaborn
column_name = 'column_of_interest'

# Histogram with Seaborn
sns.histplot(df[column_name], bins=30, kde=True)
plt.title(f'Histogram of {column_name}')
# labels are often auto-generated by Seaborn
plt.show()

# Scatter plot with regression line
sns.scatterplot(x='x_column', y='y_column', data=df)
sns.regplot(x='x_column', y='y_column', data=df, scatter=False, color='
    red')
plt.title('Scatter Plot with Regression Line')
plt.xlabel('x_column')
plt.ylabel('y_column')
plt.show()

# Pairplot for visualizing pairwise relationships in a dataset
sns.pairplot(df)
plt.show()
```

1.3 RDKit

RDKit is an open-source toolkit for cheminformatics that supports various functionalities for chemical informatics, such as reading and writing molecular data, calculating molecular properties, performing substructure searches, and generating 2D and 3D molecular representations. It is widely used in computational chemistry and bioinformatics for drug discovery and molecular modeling tasks.

```python
# Import RDKit modules
from rdkit import Chem
from rdkit.Chem import Descriptors
from rdkit.Chem import Draw

# Create a molecule from a SMILES string
smiles = 'CCO'  # Ethanol (hydrogens are omitted)
molecule = Chem.MolFromSmiles(smiles)

# Print formular, calculate weight
print("Molecular Formula:", Chem.rdMolDescriptors.CalcMolFormula(molecule))
mol_weight = Descriptors.MolWt(molecule)
print("Molecular Weight:", mol_weight)

# Draw the molecule
Draw.MolToImage(molecule)
# In a notebook one can simply do
display(molecule)

# Convert a molecule to a SMILES string
smiles_back = Chem.MolToSmiles(molecule)

# Calculate other properties, based on heuristics
num_h_donors = Descriptors.NumHDonors(molecule)
num_h_acceptors = Descriptors.NumHAcceptors(molecule)
logp = Descriptors.MolLogP(molecule)

# Perform a substructure search
substructure = Chem.MolFromSmarts('CO')  # Look for an alcohol group
matches = molecule.GetSubstructMatches(substructure)
```

2 The Problems

2.1 Inspect the Column Names

Which is not a column in `solubility_dataset.csv`, multiple answers allowed? (1 P)

(a) NumBonds

(b) Solubility

(c) InChIKey

(d) SELFIES

(e) SMILES

2.2 Size of a DataFrame

How many rows does the table (the CSV-file) have? (1 P)

2.3 Basic Stats

1. Compute the mean and standard deviation of the column `Solubility`. Round your answer to 2 digits after the decimal point. (each 0.5 P)

2. Compute the covariance of `Solubility` and `MolWt`. Round your answer to 2 digits after the decimal point. (1 P)
   The general formula is

   $$\mathrm{cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]$$

   the normalized one

   $$\mathrm{cov}(X,Y) = \frac{\mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])]}{\sqrt{\mathbb{E}[(X - \mathbb{E}[X])^2]\mathbb{E}[(Y - \mathbb{E}[Y])^2]}}$$

3. Compute the covariance of the number of heavy atoms and the number of all atoms (including H) for the data set (hint: you will need to use RDKit for this). Round your result up to two digits after the decimal point. (1 P)

2.4 Getting the Index of a Value

1. The column `MolWt` gives the Molar weight of each compound and `RingCount` the total number of rings (aliphatic and aromatic). How many rings are in the haviest compound? (0.5 P)

2. The column `NumValenceElectrons` contains, as the name suggests, the number of valence electrons of each compound. Find the InChIKey of compound with the fewest valence electrons. (0.5 P)

2.5 Display Lewis Structure

Display the Lewis structure of the 56th molecule in the table (index 55, remember Python starts counting at 0). (1 P)

2.6 Plotting

1. Make a histogram of the data in the column `MolLogP`, you can use `matplotlib` or `seaborn`. (1 P)

2. Make a scatter plot of `Solubility` (x-axis) and `MolLogP` (y-axis). (1 P)

2.7 Searching Strings

How many compounds contain at least one zink ion/atom (hint: use the SMILES)? (1 P)