MAS2008 ASSIGNMENT: EXOPLANETS DATA ANALYSIS

1. Introduction

This assignment will ask you to analyse a dataset about exoplanets (i.e. planets orbiting stars other than the sun). You should visit the following URL:

`https://strickland1.org/courses/MAS2008/assignments/exoplanets.php`

From there you can download the following files:

- The main dataset is `exoplanets.csv`. This can be imported directly by `pandas` without any pre-processing.
- The file `exoplanets_metadata.txt` contains some information about the meaning of the columns in the dataset.
- The file `milky_way.json` contains coordinates of an outline of the Milky Way.
- The file `colours.txt` contains a list of hex codes for 20 different colours.

2. Comments on the dataset

- The column `pl_name` contains the name of a planet. For some planets there are a number of different records, each with their own row; these might indicate detections at different times by different telescopes using different methods of analysis, for example.
- Each planet is attached to a star system, which may consist of a single star, or a group of two, three or four stars orbiting closely around each other. The column `hostname` gives the name of the star system.
- The number of stars in the relevant star system is given in the column `sy_snum`, and the number of planets is in the column `sy_pnum`.
- Positions of stars in the sky are often specified in terms of angles called the right ascension and declination, which are usually measured in degrees. These are recorded in the columns `ra` and `dec`. (The columns `rastr` and `decstr` contain the same information in a different format which is less useful for this assignment.)
- Most numerical columns have units, which are specified in square brackets in the file `exoplanets_metadata.txt`. For example, the line
  `# COLUMN pl_orbper: Orbital Period [days]`
  indicates that the value in column `pl_orbper` is the number of days that it takes for the planet to orbit around its star. Other units used include `au` (approximately 1.5×10^{11} metres, the distance from the earth to the sun) and `pc` (parsec, approximately 3×10^{16} metres). Some quantities are measured relative to the earth, the sun, or Jupiter. For example, the line
  `# COLUMN pl_radj: Planet Radius [Jupiter Radius]`
  means that the entry in column `pl_radj` is the radius of the relevant planet divided by the radius of Jupiter.
- For each exoplanet there is an angle i called the *inclination*, between the line of sight from the earth and the normal to the orbital plane. The most common method of finding exoplanets will only detect examples where i is close to $90°$, so $\sin(i)$ is close to 1. Some common methods for finding the mass m of the planet actually find $m\sin(i)$, which will usually be close to m but not always. The column `pl_bmasse` contains values which are measures of m in some cases and measures of $m\sin(i)$ in some other cases. The column `pl_bmassprov` contains information about which is which.
- The column `discoverymethod` contains a string describing the method used to detect the planet.
- The column `disc_facility` specifies the facility that detected the planet. This is typically the name of a telescope or an observatory or an astronomical satellite.

You can read the file `milky_way.json` like this:

```
with open('milky_way.json') as f:
    milky_way = json.load(f)
```

Now `milky_way` will be a list of length two, like `[a,b]`. Here `a` specifies one edge of the Milky Way, and `b` specifies the other edge. In more detail, `a` is a list of lists of length two, each of which contains the right ascension and declination of a point on the upper edge of the Milky Way. You can use this to add an outline of the Milky Way to any plots that you might draw.

For the file `colours.txt`, you should just cut and paste the contents into a suitable place in your code.

3. Basic tasks

3.1. **Capped masses.** Write a function `cap_masses(df)` which adds a new column called `capped_mass` to the dataframe. The value of `capped_mass` should be the same as `pl_bmasse` except that for very large planets where `pl_bmasse` is larger than 500, the value of `capped_mass` should just be 500.

The remaining tasks will assume that `cap_masses(df)` has been executed, so that the `capped_mass` column is available.

3.2. **Overall map.** Define and execute a function `overall_map(df)` which makes a map of all the planets.

- The title of the picture should be 'Exoplanet map'.
- The plot should have size 20×8, with equal scales on the two axes.
- There should be a point for each row of the dataframe, in position given by the right ascension and declination. The colour should be determined by the capped mass. (Use `df.plot.scatter()` with an appropriate value for the optional argument `c`.)
- The picture should also show the edges of the Milky Way as dashed green lines.
- You should see that there is a very dense group of points on the edge of the Milky Way near the top right of the picture, and another noticeable group further down and a little to the left in the middle of the Milky Way. Your picture should include a red circle around the first group and a blue circle around the second one (you could use `ax.add_patch()` and `plt.Circle()` for this).

Just after the plot, you should include a markdown cell explaining what the two circled groups represent. However, you might want to complete the other tasks before writing this explanation, because the reasons will become easier to understand later.

3.3. **Systems dataframe.** Write a function `analyse_systems(df)` to analyse the different star systems. The function should construct and return a new dataframe called `systems`.

- The columns should be called `name`, `number of stars` and `number of planets`.
- There should be one row for each name that appears in the `hostname` column. Each name should only appear once in the `systems` dataframe.
- The `number of stars` and `number of planets` columns should contain the numbers taken from the `sy_snum` and `sy_pnum` columns of the original dataframe.
- The systems should be sorted in order. Systems with more stars should be listed first. Among systems with the same number of stars, those with more planets should be listed first.
- The index of the dataframe should consist of integers starting from 0.

You can use the following approach: start with the original dataframe, select only the columns that you need, use the methods `drop_duplicates()`, `sort_values()` and `reset_index()` with appropriate arguments, and then rename the columns.

After defining the function `analyse_systems()` you should execute `systems = analyse_systems(df)` so that the `systems` dataframe is available for subsequent tasks.

3.4. **Facilities dataframe.** Write a function `analyse_facilities(df)` to analyse the different facilities (as listed in the `disc_facility` column) that contributed records to the dataset. The function should construct and return a new dataframe called `facilities`.

- The columns should be called `name`, `count`, `min_ra`, `max_ra`, `min_dec`, `max_dec`, `ra_range`, `dec_range` and `colour`.

- There should be one row for each facility name that appears in the `disc_facility` column, except that rows where `disc_facility` is 'Multiple Observatories' or 'Multiple Facilities' should be discarded.
- The `count` column should give the number of rows in the original dataframe corresponding to the named facility.
- The `min_ra` column should contain the minimum of the right ascension values (in the `ra` column) for rows corresponding to the named facility. The `max_ra`, `min_dec` and `max_dec` columns should follow the same pattern.
- The `ra_range` column should contain the difference between the `max_ra` and `min_ra` columns, and similarly for `dec_range`.
- The facilities should be listed in decreasing order of the `count` column.
- The `colours` column should contain colours (represented as hex code strings) taken from the file `colours.txt`. The first 19 rows in the `facilities` dataframe should have the first 19 colours from `colours.txt`. The remaining rows (which have relatively small counts) can all share the 20th colour.
- The index of the dataframe should just contain numbers starting from zero.

You will probably need to construct the new dataframe in a number of steps, some of which will involve the methods `groupby()`, `agg()`, `rename()`, `sort_values()` and `reset_index()`.

After defining the function `analyse_facilities()` you should execute `facilities = analyse_facilities(df)` so that the `facilities` dataframe is available for subsequent tasks.

Now define a function `show_facility(i)` which plots all the observations by the ith facility. Ideally the size and colour of points in this plot should reflect some information about the relevant observations.

Finally, add a `colour` column to the original dataframe `df` so that each row has the colour corresponding to its facility. (Use the same method as in the periodic table task in lab 8.)

3.5. **Insolation and equilibrium temperature.** Make a dataframe `df0` as follows:

- Start with the original dataframe `df`.
- Keep only the following columns: the equilibrium temperature `pl_eqt`, the insolation `pl_insol` and the colour.
- Keep only the rows where the equilibrium temperature is between 2000 and 2500.
- Discard any rows with missing data (represented by `NaN`).

Now make a scatter plot with `pl_eqt` on the x-axis and `pl_insol` on the y-axis, with points coloured as specified by the `colour` column. You should see that the points lie on two clearly separated curves. (I have not found out the reason for this. If you like, you could take it as a challenge to investigate the question, but that is not required for this assignment.) Either by trial and error or a more systematic method, find the equation of a straight line that runs in between the two curves. Use this to separate `df0` into two dataframes `df1` (containing points on the upper curve) and `df2` (containing points on the lower curve). Then do linear regression on these dataframes to find equations for two straight lines approximating the two curves. Make a new plot including these two lines, with the upper one in red and the lower one in blue.

Once you have done all this, combine it into a function `show_insolation(df)`. This should construct `df0`, `df1` and `df2` and generate the final plot including the two approximating lines and return the pair `(df1,df2)`. The notebook that you submit should just contain the definition of `show_insolation(df)`, you need not include any work that you did building up to that.

4. Exploration and commentary

Use pandas commands to explore the dataframe and answer three questions about it. You can choose three questions from the list below, or you can think of your own questions. You should explain your answers with reference to tables or plots produced by pandas, but these should be carefully structured to display the key points without burying the reader in vast amounts of data. For some questions you should search for (and cite) additional information from outside the dataset.

This section will carry 40% of the credit for the assignment, so you should write substantial comments that show meaningful research effort. The prompts below are merely a starting point. Each answer should be written in a separate, clearly marked section of the notebook.

(a) Which are the most important facilities contributing records to the dataset? What methods did they use, and in which years did they publish most of their results?
(b) Some facilities observe a fixed part of the sky, some of them observe the whole of the northern hemisphere (where the declination is positive) or the whole of the southern hemisphere, and some of them observe the whole sky. Which of these cases apply for the top twenty facilities?
(c) Is there a relationship between the temperature of a planet (`pl_eqt`) and the temperature of the star that it orbits (`st_teff`)? Give quantitative statistics as well as plots to explain your answer.
(d) What can you say about the relationship between the orbital period and the orbital semi-major axis? How does the picture change if we consider only planets where the orbital period is less than 100 days?
(e) How many different discovery methods are there? How does this change if we ignore methods that were used for less than 100 of the rows? For records using the 'Microlensing' method, we have no no data about the planetary radius. For records obtained by the 'Imaging' method, the average orbital period is hugely larger than for records obtained by the 'Transit' method. What other facts like this can you observe?

5. Guidance

- You may discuss the assignment with other students, but you must not copy code or text from them. You must write your own notebook in your own words based on your own understanding. You must also mention any collaboration in the acknowledgements section of your notebook, including the names of people with whom you worked.
- You may search the internet for information, but you must mention all sources that you have used in your acknowledgements, with specific URLs.
- You can use code that you find on the internet or that is given to you by an AI assistant, but you must acknowledge it. You must also ensure that the code you submit complies precisely with the notation and terminology used in this briefing document, and that the function names, arguments and return values are exactly as specified. You will probably need to modify code obtained from elsewhere to achieve this.
- All acknowledgements must appear in a separate markdown cell at the top of your notebook, with heading *Acknowledgements*.
- All nontrivial functions should have docstrings.
- For all code that implements a nonobvious algorithm, you should add detailed comments in the code to prove that you understand how the algorithm works.
- When developing your notebook, you will probably move backwards and forwards, inserting things and executing code in different places. However, before submission you should tidy up your code. Remove anything that is not needed and check that the rest can be executed in order from the top to the bottom without errors and that this generates all the required plots and prints all the required messages.
- Upload your notebook using Blackboard.
- Do not include your own name or registration number in your notebook. (Blackboard will ensure that your work is tagged with your name at the point when that becomes necessary.)