Platform: Jupyter Notebook, Python 3.9.7

The following code packages are required to be configurated.

pip install numpy pandas geopandas dbfread sympy matplotlib shapely scipy scikit-learn torch

- NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays.
- Pandas is a fast, powerful, flexible and easy to use open-source data analysis and manipulation tool.
- SymPy is a Python library for symbolic mathematics.
- SciPy provides algorithms for optimization, integration, interpolation, eigenvalue problems, algebraic equations, differential equations, statistics and many other classes of problems.
- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
- Shapely is a BSD-licensed Python package for manipulation and analysis of planar geometric objects.
- GeoPandas extends the datatypes used by pandas to allow spatial operations on geometric types, which allows for the frequent operations on the raw data in a ArcGIS format.
- Dbfread reads DBF files and returns the data as native Python data types for further processing.
  Scikit-learn is an open-source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection, model evaluation, and many other utilities.
- Torch provides Tensors and Dynamic neural networks in Python with strong GPU acceleration.

0. /Code/

The Code folder includes all the source codes and datasets, which are categorized in the following six subfolders according to their functions:

- **1_sliding-multifractal**
- **2_Box-counting**
- **3_fry**
- **4_feature-select-PA**
- **5_Kmeans**
- **6_Machine-Learning**

Wherein, **1_sliding-multifractal** is used for multifractal analyses based on sliding window; **2_Box-counting** is utilized to calculate the box-counting fractal dimensions of tungsten occurrences; **3_fry** is employed to generate Fry points of tungsten occurrences; **4_feature-select-PA** is used to create P-A plots for feature selection; **5_Kmeans** is employed in K-means clustering for feature selection; **6_Machine-Learning** is utilized to implement machine learning-based predictive modeling and model assessment, which has 8 subfolders corresponding to four machine learning algorithms trained by fractal datasets and raw datasets.

1. /Code/1_sliding-multifractal/

The multifractal calculation conducted on each of 19,5174 cells is quite time-consuming. The program is suggested to be ran in a high-configuration computer.

The folder contains 5 subfolders:

- **1_line_fauden**: to analyze features of faults
- **2_area_granite**: to analyze features of Yanshanian intrusions
- **3_area_mag**: to analyze features of magnetic anomalies

- **4_RS_Fe**: to analyze features of iron-oxide alteration
- **5_RS_QJ**: to analyze features of argillic alteration

Each subfolder includes input dataset and codes: **Code.ipynb** is the source code; **XX_YY.csv** stores the coordinate of center point of the cells, which is used for window sliding; **.shp files** are the original features under analysis; The subfolders including **1_begin, 2_clear, 3_X1, 3_Xq, 4_Dq** are cache folders used for storing temporary results during the calculation. After calculation, the multifractal indices are written into **XX_YY.csv**.

The main steps of calculation conducted by **Code.ipynb** include:

- ✓ Input .shp and XX_YY.csv
- ✓ Generate calculating window
- ✓ Calculate multifractal indices of captured features within the window
- ✓ Slide the window
- ✓ Result output to XX_YY.csv

2. /Code/2_Box-counting/

Code.ipynb is the source code; **Label_Point.shp** is the input file of occurrence points; **Range.shp** defines the areal range of box-counting analysis; **1_begin, 2_clear** are cache folders; After calculation, the analytical result is output to **/2_clear/result.png**.

The main steps of box-counting analysis include:

- ✓ Input .shp layers
- ✓ Subdivide the layer with a set of grids
- ✓ Count target features within the grids
- ✓ Calculate box-counting fractal dimensions
- ✓ Output the result to **/2_clear/result.png**.

3. /Code/3_fry/

Code.ipynb is the source code; **Label_Point.shp** is the input file of occurrence points; The program creates a **fry.shp** to store the resultant Fry points. The Fry points were then exported to drawing software to plot rose diagrams.

4. /Code/4_feature-select-PA/

The folder contains 5 subfolders:

- **1_line_fauden**: to analyze features of faults
- **2_area_granite**: to analyze features of Yanshanian intrusions
- **3_area_mag**: to analyze features of magnetic anomalies
- **4_RS_Fe**: to analyze features of iron-oxide alteration
- **5_RS_QJ**: to analyze features of argillic alteration

Each subfolder includes the source code **Code.ipynb** and ***.csv** containing the raw and fractal representation of feature values. In the **.csv**, **occu** denotes whether the cell containing tungsten occurrence (1=occurrence; 0=non-occurrence); **D0**: capacity dimension; **D1**: information dimension; **D2**: correlation dimension; **V_a**: $\Delta\alpha$, width of multifractal spectrum; **V_f**: $\Delta f(\alpha)$, height difference of multifractal spectrum; the last column is the raw representation of corresponding feature. The program creates a **Point.csv** to store the results after code running.

The main steps of the program include:

- ✓ Input feature values
- ✓ Logistic transformation
- ✓ C-A modeling to obtain threshold points

- ✓ Generating P-A points and output the information of intersection points to **Point.csv**

5. /Code/5_Kmeans/

Code.ipynb is the source code; **original_data.csv** is input files containing the feature values, wherein, **occu** denotes whether the cell containing tungsten occurrence (1=occurrence); **nondeposit** denotes whether the cell deems a non-deposit location (i.e., 1=negative sample); **fauden_***: fault-related features; **tointru_***: Yanshanian intrusion-related features; **mag_***: magnetic anomaly-related features; **RSFe_***: iron-oxide alteration-related features; **RSQJ_***: argillic alteration-related features; ***_D0**: capacity dimension; ***_D1**: information dimension; ***_D2**: correlation dimension; ***_Va**: $\Delta\alpha$, width of multifractal spectrum; ***_Vf**: $\Delta f(\alpha)$, height difference of multifractal spectrum.

The program creates two cache subfolders (**1_file and 2_deal**), and **3_Sort** is created for storing the final results of five ore-related elements, which contains **fauden.csv, mag.csv, RSFe.csv, RSQJ.csv**, and **tointru.csv**. Taking **fauden.csv** as an example, **fauden_k** records the value of K; **fauden_cluster_num** denotes the series number of the cluster; **fauden_occu** denotes the tungsten occurrences falling within the cluster; **fauden_point** denotes the total number of the points within the cluster; **fauden_occu%**=fauden_occu/total number of occurrence; **fauden_point%**= fauden_point/total number of points; **fauden_radio= auden_occu%/ auden_point%**. It is noteworthy that only those clusters capturing more than 10 occurrences are recorded in this table, in order to reduce the computation time.

The main steps of the program include：

- ✓ Input **original_data.csv**
- ✓ Set K value
- ✓ Implement K clustering
- ✓ Calculate and sort the indices within each cluster
- ✓ Output the results to /**3_Sort/**

6. /Code/6_Machine-Learning/

The folder contains 8 subfolders:

- **ANN_Fra:** ANN predictive modeling using fractal dataset
- **ANN_NoFra:** ANN predictive modeling using raw dataset
- **DT_Fra:** DT predictive modeling using fractal dataset
- **DT_NoFra:** DT predictive modeling using raw dataset
- **LR_Fra:** LR predictive modeling using fractal dataset
- **LR_NoFra:** LR predictive modeling using raw dataset
- **RF_Fra:** RF predictive modeling using fractal dataset
- **RF_NoFra:** RF predictive modeling using raw dataset

Each subfolder includes the following files and subfolders. **Code.ipynb** is the source code; **Fractal_Label.csv** is is the entire label dataset; **Fractal.csv** is raw data to be predicted; The program creates five subfolders to store results and cache files. **TrainDataset** contains 10 training datasets, **Predict** stores predictions yielded by 10 trained models; **Success** contains cache files to generate success-rate curve; **Index** output the Accuracy and Kappa index; **Fig** output prospectivity-uncertainty plots.

The main steps of the program include：

- ✓ Ten training datasets are extracted from **Fractal_Label.csv** in a 1:1 ratio based on the labels and saved to **/TrainDataset/**
- ✓ Input training datasets in **/TrainDataset/**
- ✓ Train models via 10-fold validation and grid-search parameter optimization. Assess classification performance of models by Accuracy and Kappa index, The average evaluation metrics across the ten

datasets serve as the final performance evaluation index of a model. These average evaluation metrics are stored in **/Index/data.txt**

- ✓ To predict **Fractal.csv**, generate ten sets of prediction probabilities using the ten training sets, and save them to the **/Predict/**
- ✓ The probabilities of the ten predictions are averaged and arranged in descending order to prepare for subsequent steps.
- ✓ Assess predictive efficiency of models by success-rate curve and store in **/Success/**
- ✓ Plot prospectivity-uncertainty and evaluate targeting efficiency of low-risk regions, saving them in **/Fig/**