

Debugging Notes

1,

Error:

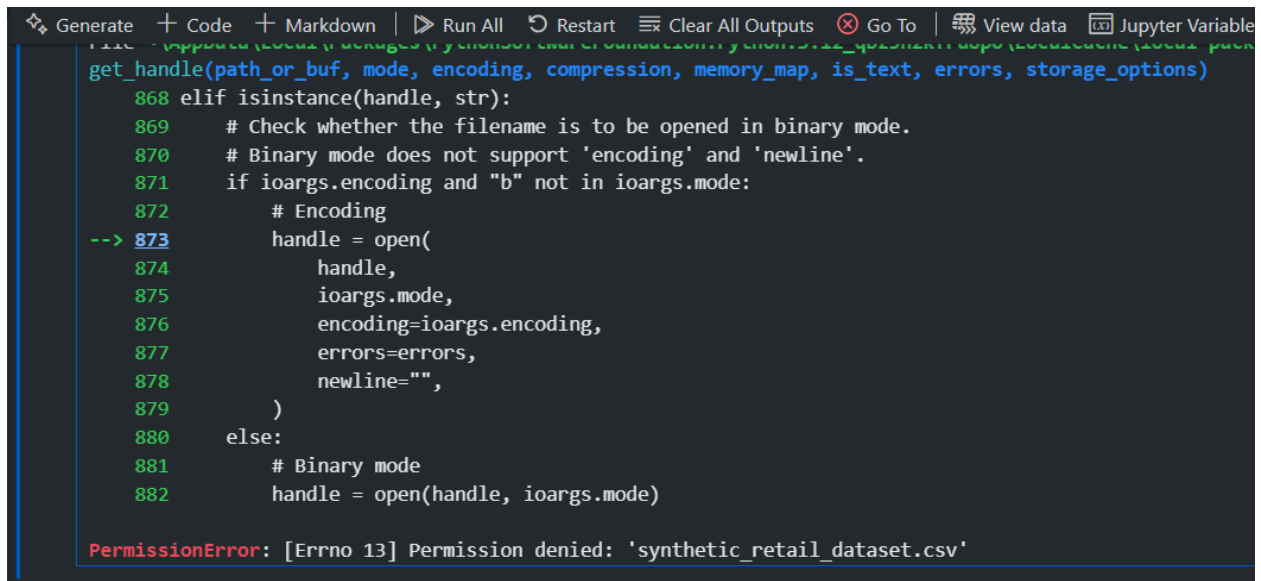
I received an error while trying to access a file.

Cause:

The error occurred because I was attempting to use a file that was already opened.

Solution:

Ensure that the file is closed before opening it again in your program, or use file handling methods (like with open(...) as f: in Python) that automatically manage closing the file.



The screenshot shows a Jupyter Notebook interface with a code cell. The code is a function `get_handle` that opens a file based on the provided arguments. The error message at the bottom of the cell is `PermissionError: [Errno 13] Permission denied: 'synthetic_retail_dataset.csv'`. The error occurs at line 873, where the file is opened in binary mode.

```
get_handle(path_or_buf, mode, encoding, compression, memory_map, is_text, errors, storage_options)
868 elif isinstance(handle, str):
869     # Check whether the filename is to be opened in binary mode.
870     # Binary mode does not support 'encoding' and 'newline'.
871     if ioargs.encoding and "b" not in ioargs.mode:
872         # Encoding
--> 873         handle = open(
874             handle,
875             ioargs.mode,
876             encoding=ioargs.encoding,
877             errors=errors,
878             newline="",
879         )
880     else:
881         # Binary mode
882         handle = open(handle, ioargs.mode)

PermissionError: [Errno 13] Permission denied: 'synthetic_retail_dataset.csv'
```

2,

Error:

While exporting the data, I used the original dataframe instead of the preprocessed one, which caused missing encoded columns.

Cause:

The original dataframe did not contain the encoded columns that were added during preprocessing.

Solution:

Always export the preprocessed dataframe that includes all necessary transformations, such as encoded columns, rather than the original raw dataframe.

```
File
~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\LocalCache\local-packages\Python312\site-packages\pandas\core\frame.py:4102, in
DataFrame.__getitem__(self, key)
    4100 if self.columns.nlevels > 1:
    4101     return self._getitem_multilevel(key)
-> 4102 indexer = self.columns.get_loc(key)
    4103 if is_integer(indexer):
    4104     indexer = [indexer]

File
~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.12_qbz5n2kfra8p0\LocalCache\local-packages\Python312\site-packages\pandas\core\indexes\base.py:3812, in
Index.get_loc(self, key)
    3807 if isinstance(casted_key, slice) or (
    3808     isinstance(casted_key, abc.Iterable)
    3809     and any(isinstance(x, slice) for x in casted_key)
    3810 ):
    3811     raise InvalidIndexError(key)
-> 3812 raise KeyError(key) from err
    3813 except TypeError:
    3814     # If we have a listlike key, _check_indexing_error will raise
    3815     # InvalidIndexError. Otherwise we fall through and re-raise
    3816     # the TypeError.
    3817     self._check_indexing_error(key)

KeyError: 'species_encoded'
```

3,

Error:

I was unable to access the cluster centers after running `kmeans_clustering_evaluate()` in my modular code.

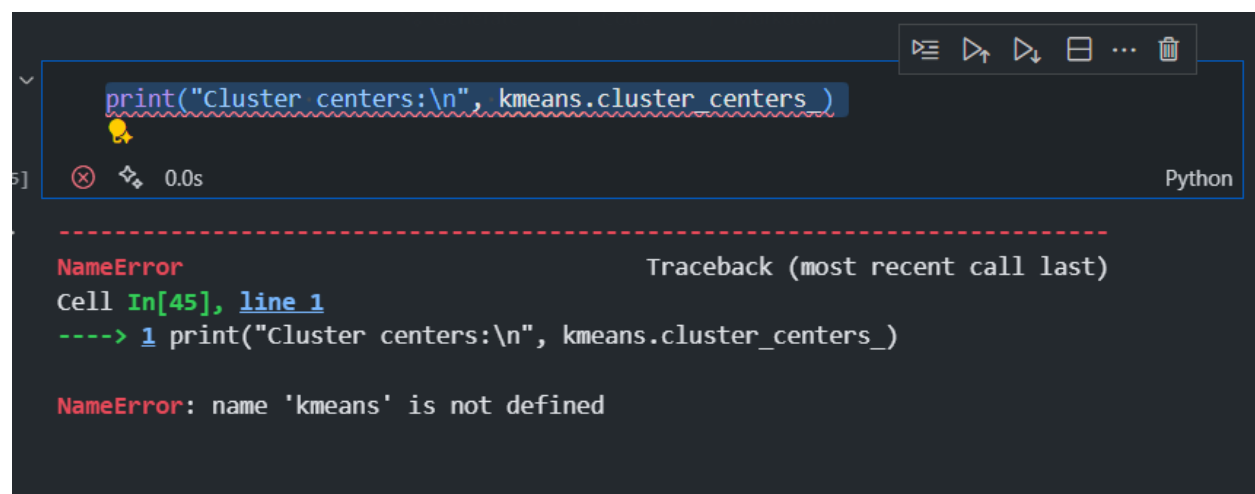
Cause:

The KMeans object was created inside the function but not returned. Currently, the function only returns clusters and ARI, so the fitted model (and its `cluster_centers_`) is inaccessible outside the function.

Solution:

Modify the function to also return the fitted KMeans model. This allows access to the cluster centers, for example:

`return clusters, ari, kmeans_model.`



The screenshot shows a Jupyter Notebook interface. At the top, there is a toolbar with icons for running, stepping through, and other cell actions. Below the toolbar, a code cell is displayed with the following code: `print("Cluster centers:\n", kmeans.cluster_centers_)`. The code is highlighted in blue. Below the code, there is a status bar showing a red 'x' icon, a yellow warning icon, and the text '0.0s' and 'Python'. Below the status bar, a traceback is shown, indicating a `NameError` with the message 'name 'kmeans' is not defined'. The traceback is titled 'Traceback (most recent call last)' and shows the error occurred in 'Cell In[45], line 1' at line 1, column 1. The error message is 'NameError: name 'kmeans' is not defined'.

```
print("Cluster centers:\n", kmeans.cluster_centers_)
```

NameError Traceback (most recent call last)
Cell In[45], line 1
----> 1 print("Cluster centers:\n", kmeans.cluster_centers_)

NameError: name 'kmeans' is not defined

4,

Error:

When running the SQL code again, I got a “table already exists” error.

Cause:

The error occurs because the schema or tables already exist. Attempting to create them again without first removing or checking them causes a conflict.

Solution:

Use DROP TABLE IF EXISTS before creating the tables to ensure that any existing tables are removed, e.g.:

```
DROP TABLE IF EXISTS fact_sales;
```

```
DROP TABLE IF EXISTS dim_customer;
```

```
DROP TABLE IF EXISTS dim_product;
```

```
DROP TABLE IF EXISTS dim_date;
```

This ensures that the tables can be recreated without errors.