In [40]:
```
import pandas as pd
import numpy as np
```

In [48]:
```
features= ["age", "workclass", "fnlwgt", "education", "education_num",
               "marital_status", "occupation", "relationship", "race", "sex",
               "capital_gain", "capital_loss", "hours_per_week", "native_country", "salary"]
df=pd.read_csv("adult.data",names=features)
df
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
Cell In[48], line 4
      1 features= ["age", "workclass", "fnlwgt", "education", "education_num",
      2                "marital_status", "occupation", "relationship", "race", "sex",
      3                "capital_gain", "capital_loss", "hours_per_week", "native_country", "salary"]
----> 4 df=pd.read_csv("adult.data",names=features)
      5 df

File ~\anaconda\lib\site-packages\pandas\util\_decorators.py:211, in deprecate_kwarg.<locals>._deprecate_kwarg.<locals>.wrapper(*args, **kwargs)
    209     else:
    210         kwargs[new_arg_name] = new_arg_value
--> 211 return func(*args, **kwargs)

File ~\anaconda\lib\site-packages\pandas\util\_decorators.py:331, in deprecate_nonkeyword_arguments.<locals>.decorate.<locals>.wrapper(*args, **kwargs)
    325 if len(args) > num_allow_args:
    326     warnings.warn(
    327         msg.format(arguments=_format_argument_list(allow_args)),
    328         FutureWarning,
    329         stacklevel=find_stack_level(),
    330     )
--> 331 return func(*args, **kwargs)

File ~\anaconda\lib\site-packages\pandas\io\parsers\readers.py:950, in read_csv(filepath_or_buffer, sep, delimiter, header, names, index_col, usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters, true_values, false_values, skipinitialspace, skiprows, skipfooter, nrows, na_values, keep_default_na, na_filter, verbose, skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col, date_parser, dayfirst, cache_dates, iterator, chunksize, compression, thousands, decimal, lineterminator, quotechar, quoting, doublequote, escapechar, comment, encoding, encoding_errors, dialect, error_bad_lines, warn_bad_lines, on_bad_lines, delim_whitespace, low_memory, memory_map, float_precision, storage_options)
    935 kwds_defaults = _refine_defaults_read(
    936     dialect,
    937     delimiter,
   (...)
    946     defaults={"delimiter": ","},
    947 )
    948 kwds.update(kwds_defaults)
--> 950 return _read(filepath_or_buffer, kwds)

File ~\anaconda\lib\site-packages\pandas\io\parsers\readers.py:605, in _read(filepath_or_buffer, kwds)
    602 _validate_names(kwds.get("names", None))
    604 # Create the parser.
--> 605 parser = TextFileReader(filepath_or_buffer, **kwds)
    607 if chunksize or iterator:
    608     return parser

File ~\anaconda\lib\site-packages\pandas\io\parsers\readers.py:1442, in TextFileReader.__init__(self, f, engine, **kwds)
   1439     self.options["has_index_names"] = kwds["has_index_names"]
   1441 self.handles: IOHandles | None = None
-> 1442 self._engine = self._make_engine(f, self.engine)

File ~\anaconda\lib\site-packages\pandas\io\parsers\readers.py:1735, in TextFileReader._make_engine(self, f, engine)
   1733     if "b" not in mode:
   1734         mode += "b"
-> 1735 self.handles = get_handle(
   1736     f,
   1737     mode,
   1738     encoding=self.options.get("encoding", None),
   1739     compression=self.options.get("compression", None),
   1740     memory_map=self.options.get("memory_map", False),
   1741     is_text=is_text,
   1742     errors=self.options.get("encoding_errors", "strict"),
   1743     storage_options=self.options.get("storage_options", None),
   1744 )
   1745 assert self.handles is not None
   1746 f = self.handles.handle

File ~\anaconda\lib\site-packages\pandas\io\common.py:856, in get_handle(path_or_buf, mode, encoding, compression, memory_map, is_text, errors, storage_options)
    851 elif isinstance(handle, str):
    852     # Check whether the filename is to be opened in binary mode.
    853     # Binary mode does not support 'encoding' and 'newline'.
    854     if ioargs.encoding and "b" not in ioargs.mode:
    855         # Encoding
--> 856         handle = open(
    857             handle,
    858             ioargs.mode,
    859             encoding=ioargs.encoding,
    860             errors=errors,
    861             newline="",
    862         )
    863     else:
    864         # Binary mode
    865         handle = open(handle, ioargs.mode)

FileNotFoundError: [Errno 2] No such file or directory: 'adult.data'
```

In [50]:
```
gender_counts = df['sex'].value_counts()
print("Number of men:", gender_counts['Male'])
print("Number of women:", gender_counts['Female'])

average_age_women = df.loc[df['sex'] == 'Female', 'age'].mean()
print("Average age of women:", average_age_women)

german_citizens_proportion = df.loc[df['native_country'] == 'Germany'].shape[0] / df.shape[0]
print("Proportion of German citizens:", german_citizens_proportion)

age_high_income_mean = df.loc[df['salary'] == '>50K', 'age'].mean()
age_high_income_std = df.loc[df['salary'] == '>50K', 'age'].std()
print("Mean age of those who receive more than 50K per year:", age_high_income_mean)
print("Standard deviation of age of those who receive more than 50K per year:", age_high_income_std)

age_low_income_mean = df.loc[df['salary'] == '<=50K', 'age'].mean()
age_low_income_std = df.loc[df['salary'] == '<=50K', 'age'].std()
print("Mean age of those who receive less than 50K per year:", age_low_income_mean)
print("Standard deviation of age of those who receive less than 50K per year:", age_low_income_std)

education_levels = ['Bachelors', 'Prof-school', 'Assoc-acdm', 'Assoc-voc', 'Masters', 'Doctorate']
high_income_education = df.loc[df['salary'] == '>50K', 'education'].isin(education_levels).all()
print("People who receive more than 50k have at least high school education:", high_income_education)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[50], line 1
----> 1 gender_counts = df['sex'].value_counts()
      2 print("Number of men:", gender_counts['Male'])
      3 print("Number of women:", gender_counts['Female'])

NameError: name 'df' is not defined
```