dgaypjmac

April 4, 2024

Lab 06

[215]:
```
# Name :- Arun M & Falak Ansari
# Reg No :- 23122110 & 23122106
# Date :- 04-04-2024
```

[216]:
```
from sklearn.model_selection import train_test_split
from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline,make_pipeline
from sklearn.feature_selection import SelectKBest,chi2
import pandas as pd
```

[217]:
```
#Read the Smaller verison of the Dataset
df = pd.read_excel('copy.xlsx')
df
```

[217]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	\
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th…	female	38.0	1	
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	
4	5	0	3					
5	6	0	3					
6	7	0	1					
7	8	0	3					
8	9	1	3					
9	10	1	2					
10	11	1	3					

```
4                            Allen, Mr. William Henry    male  35.0      0
5                                   Moran, Mr. James    male   NaN      0
6                            McCarthy, Mr. Timothy J    male  54.0      0
7                     Palsson, Master. Gosta Leonard    male   2.0      3
8   Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)  female  27.0      0
9                 Nasser, Mrs. Nicholas (Adele Achem)  female  14.0      1
10                    Sandstrom, Miss. Marguerite Rut  female   4.0      1

    Parch            Ticket     Fare Cabin Embarked
0       0         A/5 21171   7.2500   NaN        S
1       0          PC 17599  71.2833   C85        C
2       0  STON/O2. 3101282   7.9250   NaN        S
3       0            113803  53.1000  C123        S
4       0            373450   8.0500   NaN        S
5       0            330877   8.4583   NaN        Q
6       0             17463  51.8625   E46        S
7       1            349909  21.0750   NaN        S
8       2            347742  11.1333   NaN        S
9       0            237736  30.0708   NaN        C
10      1           PP 9549  16.7000    G6        S
```

[218]:
```
df.drop(columns=['PassengerId','Name','Ticket','Cabin'],inplace=True)
```

[219]:
```
#Increased the test size of the dataset
X_train,X_test,y_train,y_test = train_test_split(df.
 ↪drop(columns=['Survived']),df['Survived'],test_size=0.4,random_state=42)
```

[220]:
```
X_train
```

[220]:
```
   Pclass     Sex   Age  SibSp  Parch     Fare Embarked
1       1  female  38.0      1      0  71.2833        C
8       3  female  27.0      0      2  11.1333        S
4       3    male  35.0      0      0   8.0500        S
7       3    male   2.0      3      1  21.0750        S
3       1  female  35.0      1      0  53.1000        S
6       1    male  54.0      0      0  51.8625        S
```

[221]:
```
# imputation transformer
trf1 = ColumnTransformer([
    ('impute_age',SimpleImputer(),[2]),
    ('impute_embarked',SimpleImputer(strategy='most_frequent'),[6])
],remainder='passthrough')
# fit the preprocessor to the training data
encoded_data=trf1.fit_transform(X_train)

#Printing the encoded data
print(encoded_data)
```

```
[[38.0 'C' 1 'female' 1 0 71.2833]
 [27.0 'S' 3 'female' 0 2 11.1333]
 [35.0 'S' 3 'male' 0 0 8.05]
 [2.0 'S' 3 'male' 3 1 21.075]
 [35.0 'S' 1 'female' 1 0 53.1]
 [54.0 'S' 1 'male' 0 0 51.8625]]
```

[222]:
```
# one hot encoding
trf2 = ColumnTransformer([
    ␣
  ↪('ohe_sex_embarked',OneHotEncoder(sparse_output=False,handle_unknown='ignore'),[1,6])
],remainder='passthrough')

#Applied the OneHotEncoder to the dataframe
encoded_data2=trf2.fit_transform(X_train)

#Printed the encoded data
print(encoded_data2)
```

```
[[ 1.      0.      1.      0.      1.     38.      1.      0.     71.2833]
 [ 1.      0.      0.      1.      3.     27.      0.      2.     11.1333]
 [ 0.      1.      0.      1.      3.     35.      0.      0.      8.05  ]
 [ 0.      1.      0.      1.      3.      2.      3.      1.     21.075 ]
 [ 1.      0.      0.      1.      1.     35.      1.      0.     53.1   ]
 [ 0.      1.      0.      1.      1.     54.      0.      0.     51.8625]]
```

[223]:
```
# Scaling
trf3 = ColumnTransformer([
    ('scale',MinMaxScaler(),slice(0,10))
])
```

[224]:
```
# Feature selection
trf4 = SelectKBest(score_func=chi2,k=8)
```

[225]:
```
clf = LogisticRegression()
```

[226]:
```
pipe = Pipeline([
    ('trf1',trf1),
    ('trf2',trf2),
    ('trf3',trf3),
    ('trf4',trf4),
    ('clf',clf)
])
```

[227]:
```
# train
pipe.fit(X_train,y_train)
```

[227]:
```
Pipeline(steps=[('trf1',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('impute_age', SimpleImputer(),
                                                  [2]),
                                                 ('impute_embarked',
SimpleImputer(strategy='most_frequent'),
                                                  [6])])),
                ('trf2',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('ohe_sex_embarked',
OneHotEncoder(handle_unknown='ignore',
sparse_output=False),
                                                  [1, 6])])),
                ('trf3',
                 ColumnTransformer(transformers=[('scale', MinMaxScaler(),
                                                  slice(0, 10, None))])),
                ('trf4',
                 SelectKBest(k=8,
                             score_func=<function chi2 at 0x000001A25B6A4C10>)),
                ('clf', LogisticRegression())])
```

[228]:
```
# Display Pipeline

from sklearn import set_config
set_config(display='diagram')
```

[229]:
```
# Predict
y_pred = pipe.predict(X_test)
```

[230]:
```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

[230]:
```
0.6
```

[231]:
```
# cross validation using cross_val_score
from sklearn.model_selection import cross_val_score
cross_val_score(pipe, X_train, y_train, cv=5, scoring='accuracy').mean()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[231], line 3
      1 # cross validation using cross_val_score
      2 from sklearn.model_selection import cross_val_score
----> 3 cross_val_score(pipe, X_train, y_train, cv=5, scoring='accuracy').mean(
```

```
File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:562, in cross_val_score(estimator, X, y, groups, scoring, cv, n_jobs,␣
 ↪verbose, fit_params, pre_dispatch, error_score)
    559 # To ensure multimetric format is not supported
    560 scorer = check_scoring(estimator, scoring=scoring)
--> 562 cv_results = cross_validate(
    563     estimator=estimator,
    564     X=X,
    565     y=y,
    566     groups=groups,
    567     scoring={"score": scorer},
    568     cv=cv,
    569     n_jobs=n_jobs,
    570     verbose=verbose,
    571     fit_params=fit_params,
    572     pre_dispatch=pre_dispatch,
    573     error_score=error_score,
    574 )
    575 return cv_results["test_score"]

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\ut ls\_param_va
 ↪py:214, in validate_params.<locals>.decorator.<locals>.wrapper(*args, **kwarg )
    208 try:
    209     with config_context(
    210         skip_parameter_validation=(
    211             prefer_skip_nested_validation or global_skip_validation
    212         )
    213     ):
--> 214         return func(*args, **kwargs)
    215 except InvalidParameterError as e:
    216     # When the function is just a wrapper around an estimator, we allow
    217     # the function to delegate validation to the estimator, but we␣
 ↪replace
    218     # the name of the estimator by the name of the function in the erro
    219     # message to avoid confusion.
    220     msg = re.sub(
    221         r"parameter of \w+ must be",
    222         f"parameter of {func.__qualname__} must be",
    223         str(e),
    224     )

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:309, in cross_validate(estimator, X, y, groups, scoring, cv, n_jobs,␣
 ↪verbose, fit_params, pre_dispatch, return_train_score, return_estimator,␣
 ↪return_indices, error_score)
    306 # We clone the estimator to make sure that all the folds are
```

```
    307 # independent, and that it is pickle-able.
    308 parallel = Parallel(n_jobs=n_jobs, verbose=verbose,␣
 ↪pre_dispatch=pre_dispatch)
--> 309 results = parallel(
    310     delayed(_fit_and_score)(
    311         clone(estimator),
    312         X,
    313         y,
    314         scorers,
    315         train,
    316         test,
    317         verbose,
    318         None,
    319         fit_params,
    320         return_train_score=return_train_score,
    321         return_times=True,
    322         return_estimator=return_estimator,
    323         error_score=error_score,
    324     )
    325     for train, test in indices
    326 )
    328 _warn_or_raise_about_fit_failures(results, error_score)
    330 # For callable scoring, the return type is only know after calling. If␣
 ↪the
    331 # return type is a dictionary, the error scores can now be inserted wit
    332 # the correct key.

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\ut ls\parallel.
 ↪py:65, in Parallel.__call__(self, iterable)
     60 config = get_config()
     61 iterable_with_config = (
     62     (_with_config(delayed_func, config), args, kwargs)
     63     for delayed_func, args, kwargs in iterable
     64 )
---> 65 return super().__call__(iterable_with_config)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\joblib\par llel.
 ↪py:1863, in Parallel.__call__(self, iterable)
   1861     output = self._get_sequential_output(iterable)
   1862     next(output)
-> 1863     return output if self.return_generator else list(output)
   1865 # Let's create an ID that uniquely identifies the current call. If the
   1866 # call is interrupted early and that the same instance is immediately
   1867 # re-used, this id will be used to prevent workers that were
   1868 # concurrently finalizing a task from the previous call to run the
   1869 # callback.
```

```
   1870 with self._lock:

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\joblib\par llel.
 ↪py:1789, in Parallel._get_sequential_output(self, iterable)
   1786 yield None
   1788 # Sequentially call the tasks and yield the results.
-> 1789 for func, args, kwargs in iterable:
   1790     self.n_dispatched_batches += 1
   1791     self.n_dispatched_tasks += 1

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\ut ls\parallel.
 ↪py:61, in <genexpr>(.0)
     56 # Capture the thread-local scikit-learn configuration at the time
     57 # Parallel.__call__ is issued since the tasks can be dispatched
     58 # in a different thread depending on the backend and on the value of
     59 # pre_dispatch and n_jobs.
     60 config = get_config()
---> 61 iterable_with_config = (
     62     (_with_config(delayed_func, config), args, kwargs)
     63     for delayed_func, args, kwargs in iterable
     64 )
     65 return super().__call__(iterable_with_config)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:309, in <genexpr>(.0)
    306 # We clone the estimator to make sure that all the folds are
    307 # independent, and that it is pickle-able.
    308 parallel = Parallel(n_jobs=n_jobs, verbose=verbose,␣
 ↪pre_dispatch=pre_dispatch)
--> 309 results = parallel(
    310     delayed(_fit_and_score)(
    311         clone(estimator),
    312         X,
    313         y,
    314         scorers,
    315         train,
    316         test,
    317         verbose,
    318         None,
    319         fit_params,
    320         return_train_score=return_train_score,
    321         return_times=True,
    322         return_estimator=return_estimator,
    323         error_score=error_score,
    324     )
```

```
    325     for train, test in indices
    326 )
    328 _warn_or_raise_about_fit_failures(results, error_score)
    330 # For callable scoring, the return type is only know after calling. If␣
 ↪the
    331 # return type is a dictionary, the error scores can now be inserted wit
    332 # the correct key.

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:377, in _BaseKFold.split(self, X, y, groups)
    369 if self.n_splits > n_samples:
    370     raise ValueError(
    371         (
    372             "Cannot have number of splits n_splits={0} greater"
    373             " than the number of samples: n_samples={1}."
    374         ).format(self.n_splits, n_samples)
    375     )
--> 377 for train, test in super().split(X, y, groups):
    378     yield train, test

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:108, in BaseCrossValidator.split(self, X, y, groups)
    106 X, y, groups = indexable(X, y, groups)
    107 indices = np.arange(_num_samples(X))
--> 108 for test_index in self._iter_test_masks(X, y, groups):
    109     train_index = indices[np.logical_not(test_index)]
    110     test_index = indices[test_index]

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:770, in StratifiedKFold._iter_test_masks(self, X, y, groups)
    769 def _iter_test_masks(self, X, y=None, groups=None):
--> 770     test_folds = self._make_test_folds(X, y)
    771     for i in range(self.n_splits):
    772         yield test_folds == i

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo el_selection\
 ↪py:732, in StratifiedKFold._make_test_folds(self, X, y)
    730 min_groups = np.min(y_counts)
    731 if np.all(self.n_splits > y_counts):
--> 732     raise ValueError(
    733         "n_splits=%d cannot be greater than the"
    734         " number of members in each class." % (self.n_splits)
    735     )
    736 if self.n_splits > min_groups:
    737     warnings.warn(
```

```
    738         "The least populated class in y has only %d"
    739         " members, which is less than n_splits=%d."
    740         % (min_groups, self.n_splits),
    741         UserWarning,
    742     )

ValueError: n_splits=5 cannot be greater than the number of members in each␣
 ↪class.
```

[ ]:
```
# gridsearchcv
param_grid = {
    "clf__penalty": ['l1', 'l2'],
    "clf__C": [0.001, 0.01, 0.1, 1, 10, 100]
}
```

[ ]:
```
from sklearn.model_selection import GridSearchCV
grid = GridSearchCV(pipe, param_grid, cv=5, scoring='accuracy')
grid.fit(X_train, y_train)
```

```
C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n
2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\model_selection\_split.py:737: UserWarning: The least populated
class in y has only 3 members, which is less than n_splits=5.
  warnings.warn(
C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n
2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\model_selection\_validation.py:425: FitFailedWarning:
36 fits failed out of a total of 60.
The score on these train-test partitions for these parameters will be set to
nan.
If these failures are not expected, you can try to debug them by setting
error_score='raise'.

Below are more details about the failures:
--------------------------------------------------------------------------------
24 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 1152, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
```

```
packages\sklearn\pipeline.py", line 427, in fit
    self._final_estimator.fit(Xt, y, **fit_params_last_step)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 1152, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\linear_model\_logistic.py", line 1169, in fit
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\linear_model\_logistic.py", line 56, in _check_solver
    raise ValueError(
ValueError: Solver lbfgs supports only 'l2' or 'none' penalties, got l1 penalty.

--------------------------------------------------------------------------------
12 fits failed with the following error:
Traceback (most recent call last):
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\model_selection\_validation.py", line 729, in _fit_and_score
    estimator.fit(X_train, y_train, **fit_params)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 1152, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\pipeline.py", line 423, in fit
    Xt = self._fit(X, y, **fit_params_steps)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\pipeline.py", line 377, in _fit
    X, fitted_transformer = fit_transform_one_cached(
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\joblib\memory.py", line 353, in __call__
    return self.func(*args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\pipeline.py", line 957, in _fit_transform_one
    res = transformer.fit_transform(X, y, **fit_params)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\utils\_set_output.py", line 157, in wrapped
    data_to_wrap = f(self, X, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
```

```
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 1152, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\compose\_column_transformer.py", line 754, in fit_transform
    result = self._fit_transform(X, y, _fit_transform_one)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\compose\_column_transformer.py", line 681, in _fit_transform
    return Parallel(n_jobs=self.n_jobs)(
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\utils\parallel.py", line 65, in __call__
    return super().__call__(iterable_with_config)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\joblib\parallel.py", line 1863, in __call__
    return output if self.return_generator else list(output)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\joblib\parallel.py", line 1792, in _get_sequential_output
    res = func(*args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\utils\parallel.py", line 127, in __call__
    return self.function(*args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\pipeline.py", line 957, in _fit_transform_one
    res = transformer.fit_transform(X, y, **fit_params)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\utils\_set_output.py", line 157, in wrapped
    data_to_wrap = f(self, X, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 919, in fit_transform
    return self.fit(X, y, **fit_params).transform(X)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\preprocessing\_data.py", line 435, in fit
    return self.partial_fit(X, y)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 1152, in wrapper
    return fit_method(estimator, *args, **kwargs)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
```

```
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\preprocessing\_data.py", line 473, in partial_fit
    X = self._validate_data(
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\base.py", line 605, in _validate_data
    out = check_array(X, input_name="X", **check_params)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\utils\validation.py", line 915, in check_array
    array = _asarray_with_order(array, order=order, dtype=dtype, xp=xp)
  File "C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\utils\_array_api.py", line 380, in _asarray_with_order
    array = numpy.asarray(array, order=order, dtype=dtype)
ValueError: could not convert string to float: 'female'

  warnings.warn(some_fits_failed_message, FitFailedWarning)
C:\Users\arunp\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n
2kfra8p0\LocalCache\local-packages\Python310\site-
packages\sklearn\model_selection\_search.py:979: UserWarning: One or more of the
test scores are non-finite: [nan nan nan nan nan nan nan nan nan nan nan nan]
  warnings.warn(
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[56], line 3
      1 from sklearn.model_selection import GridSearchCV
      2 grid = GridSearchCV(pipe, param_grid, cv=5, scoring='accuracy')
----> 3 grid.fit(X_train, y_train)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\ba
 ↪py:1152, in _fit_context.<locals>.decorator.<locals>.wrapper(estimator, *args
 ↪**kwargs)
   1145     estimator._validate_params()
   1147 with config_context(
   1148     skip_parameter_validation=(
   1149         prefer_skip_nested_validation or global_skip_validation
   1150     )
   1151 ):
-> 1152     return fit_method(estimator, *args, **kwargs)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\mo
 ↪py:936, in BaseSearchCV.fit(self, X, y, groups, **fit_params)
    934 refit_start_time = time.time()
    935 if y is not None:
```

```
--> 936     self.best_estimator_.fit(X, y, **fit_params)
    937 else:
    938     self.best_estimator_.fit(X, **fit_params)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\ba e.
 ↪py:1152, in _fit_context.<locals>.decorator.<locals>.wrapper(estimator, *args
 ↪**kwargs)
   1145     estimator._validate_params()
   1147 with config_context(
   1148     skip_parameter_validation=(
   1149         prefer_skip_nested_validation or global_skip_validation
   1150     )
   1151 ):
-> 1152     return fit_method(estimator, *args, **kwargs)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\pi eline.
 ↪py:427, in Pipeline.fit(self, X, y, **fit_params)
    425     if self._final_estimator != "passthrough":
    426         fit_params_last_step = fit_params_steps[self.steps[-1][0]]
--> 427         self._final_estimator.fit(Xt, y, **fit_params_last_step)
    429 return self

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\ba e.
 ↪py:1152, in _fit_context.<locals>.decorator.<locals>.wrapper(estimator, *args
 ↪**kwargs)
   1145     estimator._validate_params()
   1147 with config_context(
   1148     skip_parameter_validation=(
   1149         prefer_skip_nested_validation or global_skip_validation
   1150     )
   1151 ):
-> 1152     return fit_method(estimator, *args, **kwargs)

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\li ear_model\_lo
 ↪py:1169, in LogisticRegression.fit(self, X, y, sample_weight)
   1139 @_fit_context(prefer_skip_nested_validation=True)
   1140 def fit(self, X, y, sample_weight=None):
   1141     """
   1142     Fit the model according to the given training data.
   1143
   (…)
   1167     The SAGA solver supports both float64 and float32 bit arrays.
   1168     """
-> 1169     solver = _check_solver(self.solver, self.penalty, self.dual)
   1171     if self.penalty != "elasticnet" and self.l1_ratio is not None:
```

```
   1172         warnings.warn(
   1173             "l1_ratio parameter is only used when penalty is "
   1174             "'elasticnet'. Got "
   1175             "(penalty={})".format(self.penalty)
   1176         )

File ~\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.
 ↪10_qbz5n2kfra8p0\LocalCache\local-packages\Python310\site-packages\sklearn\li ear_model\_lo
 ↪py:56, in _check_solver(solver, penalty, dual)
     53 def _check_solver(solver, penalty, dual):
     54     # TODO(1.4): Remove "none" option
     55     if solver not in ["liblinear", "saga"] and penalty not in ("l2",␣
 ↪"none", None):
---> 56         raise ValueError(
     57             "Solver %s supports only 'l2' or 'none' penalties, got %s␣
 ↪penalty."
     58             % (solver, penalty)
     59         )
     60     if solver != "liblinear" and dual:
     61         raise ValueError(
     62             "Solver %s supports only dual=False, got dual=%s" % (solver ␣
 ↪dual)
     63         )

ValueError: Solver lbfgs supports only 'l2' or 'none' penalties, got l1 penalty
```

[ ]:
```
grid.best_score_
```

[ ]:
```
nan
```

[ ]:
```
grid.best_params_
```

[ ]:
```
{'clf__C': 0.001, 'clf__penalty': 'l1'}
```

[ ]:
```
# export
import pickle
pickle.dump(pipe,open('lr_pile.pkl','wb'))
```