SloanDigitalSkySurvey

September 30, 2018

1 Sloan Digital Sky Survey

```
In [79]: from fastai.imports import *
         from fastai.structured import *

         from pandas_summary import DataFrameSummary
         from sklearn.ensemble import RandomForestClassifier
         from IPython.display import display

         from sklearn import metrics
```

```
In [80]: # set path to data
         PATH = "/home/chris/Datasets/SkySurvey/"
```

```
In [81]: # take a look to make sure it's there
         !ls {PATH}
```

```
'Skyserver_SQL2_27_2018 6_51_39 PM.csv.zip'   Train.csv
```

```
In [82]: # create a pandas df for our data
         df_raw = pd.read_csv(f'{PATH}Train.csv', low_memory=False)
```

```
In [83]: df_raw.head()
```

```
Out[83]:           objid          ra       dec         u         g         r         i  \
         0  1.237650e+18  183.531326  0.089693  19.47406  17.04240  15.94699  15.50342
         1  1.237650e+18  183.598371  0.135285  18.66280  17.21449  16.67637  16.48922
         2  1.237650e+18  183.680207  0.126185  19.38298  18.19169  17.47428  17.08732
         3  1.237650e+18  183.870529  0.049911  17.76536  16.60272  16.16116  15.98233
         4  1.237650e+18  183.883288  0.102557  17.55025  16.26342  16.43869  16.55492

                   z  run  rerun  camcol  field     specobjid   class  redshift  plate  \
         0  15.22531  752    301       4    267  3.722360e+18    STAR -0.000009   3306
         1  16.39150  752    301       4    267  3.638140e+17    STAR -0.000055    323
         2  16.80125  752    301       4    268  3.232740e+17  GALAXY  0.123111    287
         3  15.90438  752    301       4    269  3.722370e+18    STAR -0.000111   3306
         4  16.61326  752    301       4    269  3.722370e+18    STAR  0.000590   3306
```

```
        mjd  fiberid
0     54922      491
1     51615      541
2     52023      513
3     54922      510
4     54922      512
```

```
In [84]: df_raw.dtypes
```

```
Out[84]: objid        float64
         ra           float64
         dec          float64
         u            float64
         g            float64
         r            float64
         i            float64
         z            float64
         run            int64
         rerun          int64
         camcol         int64
         field          int64
         specobjid    float64
         class         object
         redshift     float64
         plate          int64
         mjd            int64
         fiberid        int64
         dtype: object
```

```
In [85]: # our data has 10,000 rows and 18 columns
         df_raw.shape
```

```
Out[85]: (10000, 18)
```

```
In [86]: #df_raw = df_raw.drop('objid', axis=1)
         df = train_cats(df_raw)
```

```
In [87]: # GALAXY=0 QSO=1 STAR=2
         df_raw['class'].cat.categories
```

```
Out[87]: Index(['GALAXY', 'QSO', 'STAR'], dtype='object')
```

```
In [88]: # I chose to show 20 so we can see every type of object
         df_raw['class'].cat.codes.head(20)
```

```
Out[88]: 0     2
         1     2
         2     0
         3     2
         4     2
         5     2
         6     0
         7     2
         8     2
         9     0
         10    2
         11    2
         12    2
         13    2
         14    0
         15    1
         16    2
         17    1
         18    2
         19    0
         dtype: int8
```

```
In [89]: # split our independent and dependent variables
         X_train = df_raw.drop(['class'], axis=1)
         y_train = df_raw['class']

         # instantiate our classifier
         m = RandomForestClassifier(n_jobs=-1)

         # fit our data
         m.fit(X_train, y_train)
```

```
Out[89]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                     max_depth=None, max_features='auto', max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=-1,
                     oob_score=False, random_state=None, verbose=0,
                     warm_start=False)
```

```
In [90]: # Let's see our training accuracy
         m.score(X_train, y_train)
```

```
Out[90]: 0.9983
```

```
In [91]: # possible classes
         m.classes_
```

```
Out[91]: array(['GALAXY', 'QSO', 'STAR'], dtype=object)
```

```
In [92]: # print this out to compare with processed df
         y_train[:5]
```

```
Out[92]: 0      STAR
         1      STAR
         2    GALAXY
         3      STAR
         4      STAR
         Name: class, dtype: category
         Categories (3, object): [GALAXY < QSO < STAR]
```

```
In [93]: # process dataframe
         df, y, nas = proc_df(df_raw, 'class')
```

```
In [94]: # our vectorized dependent variable
         y[:5]
```

```
Out[94]: array([2, 2, 0, 2, 2], dtype=int8)
```

```
In [95]: df_raw.head()
```

```
Out[95]:           objid          ra       dec         u         g         r         i  \
         0  1.237650e+18  183.531326  0.089693  19.47406  17.04240  15.94699  15.50342
         1  1.237650e+18  183.598371  0.135285  18.66280  17.21449  16.67637  16.48922
         2  1.237650e+18  183.680207  0.126185  19.38298  18.19169  17.47428  17.08732
         3  1.237650e+18  183.870529  0.049911  17.76536  16.60272  16.16116  15.98233
         4  1.237650e+18  183.883288  0.102557  17.55025  16.26342  16.43869  16.55492

                   z  run  rerun  camcol  field     specobjid   class  redshift  plate  \
         0  15.22531  752    301       4    267  3.722360e+18    STAR -0.000009   3306
         1  16.39150  752    301       4    267  3.638140e+17    STAR -0.000055    323
         2  16.80125  752    301       4    268  3.232740e+17  GALAXY  0.123111    287
         3  15.90438  752    301       4    269  3.722370e+18    STAR -0.000111   3306
         4  16.61326  752    301       4    269  3.722370e+18    STAR  0.000590   3306

              mjd  fiberid
         0  54922      491
         1  51615      541
         2  52023      513
         3  54922      510
         4  54922      512
```

```
In [96]: # this function will split the dataframe 80% train 20% valid
         def split_vals(a,n):
             return a[:n].copy(), a[n:].copy()

         n_valid = 2000
         n_train = len(df_raw)-n_valid

         # split the data initially
         raw_train, raw_valid = split_vals(df_raw, n_train)
         # now split each of those
         X_train, X_valid = split_vals(df, n_train)
         y_train, y_valid = split_vals(y, n_train)

         X_train.shape, y_train.shape, X_valid.shape, y_valid.shape

Out[96]: ((8000, 17), (8000,), (2000, 17), (2000,))

In [97]: # y_train took on traits of y
         y_train

Out[97]: array([2, 2, 0, ..., 0, 1, 0], dtype=int8)

In [98]: def print_score(m):
             res = [m.score(X_train, y_train), m.score(X_train, y_train)]
             if hasattr(m, 'oob_score_'):
                 res.append(m.oob_score_)
             print(res)

In [99]: m = RandomForestClassifier(n_jobs=-1)
         %time m.fit(X_train, y_train)
         print_score(m)

CPU times: user 294 ms, sys: 8.05 ms, total: 302 ms
Wall time: 243 ms
[0.99825, 0.99825]


In [100]: # possible classes after changing array
          m.classes_

Out[100]: array([0, 1, 2], dtype=int8)

In [101]: m = RandomForestClassifier(n_estimators=40, n_jobs=-1, oob_score=True)
          m.fit(X_train, y_train)
          print_score(m)

[0.99975, 0.99975, 0.99]


In [102]: m.predict([[1.237650e+18,130.1993148,51.00089591,18.68415,17.78766,17.51412,17.41314

Out[102]: array([2], dtype=int8)

In [103]: m.classes_

Out[103]: array([0, 1, 2], dtype=int8)

In [104]: m.feature_importances_
```

```
Out[104]: array([0.     , 0.00543, 0.0056 , 0.01543, 0.04344, 0.03159, 0.06321, 0.04574, 0.003
                 0.00374, 0.11626, 0.47978, 0.08179, 0.09869, 0.00452])
```

```
In [105]: X_train.head()
```

```
Out[105]:           objid          ra       dec         u         g         r         i  \
          0  1.237650e+18  183.531326  0.089693  19.47406  17.04240  15.94699  15.50342
          1  1.237650e+18  183.598371  0.135285  18.66280  17.21449  16.67637  16.48922
          2  1.237650e+18  183.680207  0.126185  19.38298  18.19169  17.47428  17.08732
          3  1.237650e+18  183.870529  0.049911  17.76536  16.60272  16.16116  15.98233
          4  1.237650e+18  183.883288  0.102557  17.55025  16.26342  16.43869  16.55492

                    z  run  rerun  camcol  field     specobjid  redshift  plate    mjd  \
          0  15.22531  752    301       4    267  3.722360e+18 -0.000009   3306  54922
          1  16.39150  752    301       4    267  3.638140e+17 -0.000055    323  51615
          2  16.80125  752    301       4    268  3.232740e+17  0.123111    287  52023
          3  15.90438  752    301       4    269  3.722370e+18 -0.000111   3306  54922
          4  16.61326  752    301       4    269  3.722370e+18  0.000590   3306  54922

             fiberid
          0      491
          1      541
          2      513
          3      510
          4      512
```

```
In [106]: X_train = X_train.drop('rerun', axis=1)
          X_valid = X_valid.drop('rerun', axis=1)
          m = RandomForestClassifier(n_estimators=40, n_jobs=-1, oob_score=True)
          m.fit(X_train, y_train)
```

```
Out[106]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=40, n_jobs=-1,
                      oob_score=True, random_state=None, verbose=0, warm_start=False)
```

```
In [107]: print_score(m)
```

```
[0.99975, 0.99975, 0.987875]
```

```
In [108]: X_train = X_train.drop('objid', axis=1)
          X_vaild = X_valid.drop('objid', axis=1)

          m = RandomForestClassifier(n_estimators=40, n_jobs=-1, oob_score=True)
          m.fit(X_train, y_train)
```

```
Out[108]: RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=40, n_jobs=-1,
                      oob_score=True, random_state=None, verbose=0, warm_start=False)
```

```
In [109]: print_score(m)
```

```
[0.999875, 0.999875, 0.988375]
```

```
In [110]: m.feature_importances_
```

```
Out[110]: array([0.00658, 0.00705, 0.02235, 0.04421, 0.04566, 0.05819, 0.05578, 0.00469, 0.0012
                 0.42183, 0.10682, 0.12361, 0.00694])
```

```
In [111]: X_train.head()
```

```
Out[111]:            ra       dec         u         g         r         i         z  \
          0  183.531326  0.089693  19.47406  17.04240  15.94699  15.50342  15.22531
          1  183.598371  0.135285  18.66280  17.21449  16.67637  16.48922  16.39150
          2  183.680207  0.126185  19.38298  18.19169  17.47428  17.08732  16.80125
          3  183.870529  0.049911  17.76536  16.60272  16.16116  15.98233  15.90438
          4  183.883288  0.102557  17.55025  16.26342  16.43869  16.55492  16.61326

             run  camcol  field     specobjid  redshift  plate    mjd  fiberid
          0  752       4    267  3.722360e+18 -0.000009   3306  54922      491
          1  752       4    267  3.638140e+17 -0.000055    323  51615      541
          2  752       4    268  3.232740e+17  0.123111    287  52023      513
          3  752       4    269  3.722370e+18 -0.000111   3306  54922      510
          4  752       4    269  3.722370e+18  0.000590   3306  54922      512
```

```
In [112]: m.oob_score_
```

```
Out[112]: 0.988375
```