Brian_Reppeto_DSC550_Week_9

May 12, 2024

0.0.1 DSC 550 Week :

Activity 9.2

Author: Brian Reppeto 5/6/2024

[88]:
```
# import libraries

import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
```

[89]:
```
# load the dataset

data=pd.read_csv('Loan_Train.csv')
```

[90]:
```
# head the data

data.head(15)
```

[90]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed
0	LP001002	Male	No	0	Graduate	No
1	LP001003	Male	Yes	1	Graduate	No
2	LP001005	Male	Yes	0	Graduate	Yes
3	LP001006	Male	Yes	0	Not Graduate	No
4	LP001008	Male	No	0	Graduate	No
5	LP001011	Male	Yes	2	Graduate	Yes
6	LP001013	Male	Yes	0	Not Graduate	No
7	LP001014	Male	Yes	3+	Graduate	No
8	LP001018	Male	Yes	2	Graduate	No
9	LP001020	Male	Yes	1	Graduate	No
10	LP001024	Male	Yes	2	Graduate	No
11	LP001027	Male	Yes	2	Graduate	NaN
12	LP001028	Male	Yes	2	Graduate	No

```
13  LP001029   Male      No          0      Graduate            No
14  LP001030   Male     Yes          2      Graduate            No

    ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0              5849                0.0         NaN             360.0
1              4583             1508.0       128.0             360.0
2              3000                0.0        66.0             360.0
3              2583             2358.0       120.0             360.0
4              6000                0.0       141.0             360.0
5              5417             4196.0       267.0             360.0
6              2333             1516.0        95.0             360.0
7              3036             2504.0       158.0             360.0
8              4006             1526.0       168.0             360.0
9             12841            10968.0       349.0             360.0
10             3200              700.0        70.0             360.0
11             2500             1840.0       109.0             360.0
12             3073             8106.0       200.0             360.0
13             1853             2840.0       114.0             360.0
14             1299             1086.0        17.0             120.0

    Credit_History Property_Area Loan_Status
0              1.0         Urban           Y
1              1.0         Rural           N
2              1.0         Urban           Y
3              1.0         Urban           Y
4              1.0         Urban           Y
5              1.0         Urban           Y
6              1.0         Urban           Y
7              0.0     Semiurban           N
8              1.0         Urban           Y
9              1.0     Semiurban           N
10             1.0         Urban           Y
11             1.0         Urban           Y
12             1.0         Urban           Y
13             1.0         Rural           N
14             1.0         Urban           Y
```

[91]:
```
# data types

data.dtypes
```

[91]:
```
Loan_ID              object
Gender               object
Married              object
Dependents           object
Education            object
Self_Employed        object
```

```
ApplicantIncome        int64
CoapplicantIncome    float64
LoanAmount           float64
Loan_Amount_Term     float64
Credit_History       float64
Property_Area         object
Loan_Status           object
dtype: object
```

[92]:
```
# data shape

data.shape
```

[92]:
```
(614, 13)
```

[93]:
```
# drop the loan_id

data.drop ('Loan_ID', axis=1, inplace=True)
```

[94]:
```
# check drop

data.head()
```

[94]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

[95]:
```
# drop any rows with missing data

data.dropna(inplace=True)
```

[96]:
```
# head new DF

data.head(15)
```

[96]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y
5	Male	Yes	2	Graduate	Yes	5417	4196.0	267.0	360.0	1.0	Urban	Y
6	Male	Yes	0	Not Graduate	No	2333	1516.0	95.0	360.0	1.0	Urban	Y
7	Male	Yes	3+	Graduate	No	3036	2504.0	158.0	360.0	0.0	Semiurban	N
8	Male	Yes	2	Graduate	No	4006	1526.0	168.0	360.0	1.0	Urban	Y
9	Male	Yes	1	Graduate	No	12841	10968.0	349.0	360.0	1.0		
10	Male	Yes	2	Graduate	No	3200	700.0	70.0	360.0	1.0		
12	Male	Yes	2	Graduate	No	3073	8106.0	200.0	360.0	1.0		
13	Male	No	0	Graduate	No	1853	2840.0	114.0	360.0	1.0		
14	Male	Yes	2	Graduate	No	1299	1086.0	17.0	120.0	1.0		
15	Male	No	0	Graduate	No	4950	0.0	125.0	360.0	1.0		
17	Female	No	0	Graduate	No	3510	0.0	76.0	360.0	0.0		

```
9      Semiurban           N
10         Urban           Y
12         Urban           Y
13         Rural           N
14         Urban           Y
15         Urban           Y
17         Urban           N
```

[97]:
```
# get shape after drops

data.shape
```

[97]:
```
(480, 12)
```

[98]:
```
# convert categorical features into dummy variables

data=pd.get_dummies(data, drop_first=True)
```

[99]:
```
# head data

data.head(15)
```

[99]:
```
    ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
1              4583             1508.0       128.0             360.0
2              3000                0.0        66.0             360.0
3              2583             2358.0       120.0             360.0
4              6000                0.0       141.0             360.0
5              5417             4196.0       267.0             360.0
6              2333             1516.0        95.0             360.0
7              3036             2504.0       158.0             360.0
8              4006             1526.0       168.0             360.0
9             12841            10968.0       349.0             360.0
10             3200              700.0        70.0             360.0
12             3073             8106.0       200.0             360.0
13             1853             2840.0       114.0             360.0
14             1299             1086.0        17.0             120.0
15             4950                0.0       125.0             360.0
17             3510                0.0        76.0             360.0

    Credit_History  Gender_Male  Married_Yes  Dependents_1  Dependents_2  \
1              1.0         True         True          True         False
2              1.0         True         True         False         False
3              1.0         True         True         False         False
4              1.0         True        False         False         False
5              1.0         True         True         False          True
6              1.0         True         True         False         False
7              0.0         True         True         False         False
```

```
8              1.0         True          True         False          True
9              1.0         True          True          True         False
10             1.0         True          True         False          True
12             1.0         True          True         False          True
13             1.0         True         False         False         False
14             1.0         True          True         False          True
15             1.0         True         False         False         False
17             0.0        False         False         False         False

    Dependents_3+  Education_Not Graduate  Self_Employed_Yes  \
1           False                   False              False
2           False                   False               True
3           False                    True              False
4           False                   False              False
5           False                   False               True
6           False                    True              False
7            True                   False              False
8           False                   False              False
9           False                   False              False
10          False                   False              False
12          False                   False              False
13          False                   False              False
14          False                   False              False
15          False                   False              False
17          False                   False              False

    Property_Area_Semiurban  Property_Area_Urban  Loan_Status_Y
1                     False                False          False
2                     False                 True           True
3                     False                 True           True
4                     False                 True           True
5                     False                 True           True
6                     False                 True           True
7                      True                False          False
8                     False                 True           True
9                      True                False          False
10                    False                 True           True
12                    False                 True           True
13                    False                False          False
14                    False                 True           True
15                    False                 True           True
17                    False                 True          False
```

[100]:
```python
# split the data

X=data.drop('Loan_Status_Y', axis=1)
y=data['Loan_Status_Y']
```

```
X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[101]:
```
# create the pipeline

pipeline=Pipeline([('scaler', MinMaxScaler()),('clf', KNeighborsClassifier())])
```

[102]:
```
# fit the pipeline

pipeline.fit(X_train, y_train)

# evaluate the model

print("Accuracy on test set:", pipeline.score(X_test, y_test))
```

```
Accuracy on test set: 0.78125
```

[103]:
```
# define the parameter grid

param_grid = {'clf__n_neighbors': list(range(1, 11))}
```

[104]:
```
# grid search with 5-fold cross-validation

grid_search=GridSearchCV(pipeline, param_grid, cv=5)
grid_search.fit(X_train, y_train)

# best parameter and accuracy

print("Best n_neighbors:", grid_search.best_params_)
print("Best score:", grid_search.best_score_)
```

```
Best n_neighbors: {'clf__n_neighbors': 3}
Best score: 0.7423103212576898
```

[105]:
```
# best model accuracy on test set

print("Accuracy on test set:", grid_search.score(X_test, y_test))
```

```
Accuracy on test set: 0.7916666666666666
```

[106]:
```
# expanded search space

param_grid = [
    {'clf': [KNeighborsClassifier()], 'clf__n_neighbors': list(range(1, 11))},
    {'clf': [LogisticRegression(max_iter=1000)], 'clf__C': [0.01, 0.1, 1, 10,␣
 ↪100]},
    {'clf': [RandomForestClassifier()], 'clf__n_estimators': [10, 50, 100, 200]}
]
```

[107]:
```
# grid search with expanded search space

grid_search = GridSearchCV(pipeline, param_grid, cv=5, verbose=1)
grid_search.fit(X_train, y_train)

# best model and parameters

print("Best parameters:", grid_search.best_params_)
print("Best accuracy on test set:", grid_search.score(X_test, y_test))
```

```
Fitting 5 folds for each of 19 candidates, totalling 95 fits
Best parameters: {'clf': LogisticRegression(C=10, max_iter=1000), 'clf__C': 10}
Best accuracy on test set: 0.8229166666666666
```

Summary:

The project, was outlined to build and fine-tune a machine learning model to predict loan status.

Starting with a flexible pipeline that incorporated a Min-Max scaler and a placeholder for classifiers, I initially fitted a default K-Nearest Neighbors (KNN) classifier. This setup allowed me to establish a baseline accuracy for the test set.

When looking to explore the potential for improving the baseline model, I used hyperparameter tuning using GridSearchCV. The first grid search focused solely on optimizing the number of neighbors for the KNN classifier across a range of values from 1 to 10. This step aimed to find the most effective configuration for this classifier in terms of prediction accuracy.

Next, I used different types of classifiers: Logistic Regression and Random Forest, in addition to the KNN. This broader search included tuning specific parameters for each classifier type, such as regularization strength for Logistic Regression and the number of trees in the Random Forest.

Each configuration was evaluated using 5-fold cross-validation to ensure that our model's performance assessment was robust and not overly fitted to a specific part of the training data. The grid search provided us with the best-performing model and parameter set.

The project's result was a carefully tuned model that either matched or surpassed the performance of the initially fitted default classifier. This process demonstrated the effectiveness of using a dynamic and flexible machine learning pipeline for model selection and hyperparameter tuning.

[ ]: