**Finding best model and hyper parameters for sklearn digits dataset classification**

## Exercise: Machine Learning Finding Optimal Model and Hyperparameters

For digits dataset in sklearn.dataset, please try following classifiers and find out the one that gives best performance. Also find the optimal parameters for that classifier.

from sklearn import svm

from sklearn.ensemble import RandomForestClassifier

from sklearn.linear\_model import LogisticRegression

from sklearn.naive\_bayes import GaussianNB

from sklearn.naive\_bayes import MultinomialNB

from sklearn.tree import DecisionTreeClassifier

**from** sklearn **import** datasets

digits **=** datasets**.**load\_digits()

**from** sklearn **import** svm

**from** sklearn.ensemble **import** RandomForestClassifier

**from** sklearn.linear\_model **import** LogisticRegression

**from** sklearn.naive\_bayes **import** GaussianNB

**from** sklearn.naive\_bayes **import** MultinomialNB

**from** sklearn.tree **import** DecisionTreeClassifier

model\_params **=** {

'svm': {

'model': svm**.**SVC(gamma**=**'auto'),

'params' : {

'C': [1,10,20],

'kernel': ['rbf','linear']

}

},

'random\_forest': {

'model': RandomForestClassifier(),

'params' : {

'n\_estimators': [1,5,10]

}

},

'logistic\_regression' : {

'model': LogisticRegression(solver**=**'liblinear',multi\_class**=**'auto'),

'params': {

'C': [1,5,10]

}

},

'naive\_bayes\_gaussian': {

'model': GaussianNB(),

'params': {}

},

'naive\_bayes\_multinomial': {

'model': MultinomialNB(),

'params': {}

},

'decision\_tree': {

'model': DecisionTreeClassifier(),

'params': {

'criterion': ['gini','entropy'],

}

}

}

**from** sklearn.model\_selection **import** GridSearchCV

**import** pandas **as** pd

scores **=** []

**for** model\_name, mp **in** model\_params**.**items():

clf **=** GridSearchCV(mp['model'], mp['params'], cv**=**5, return\_train\_score**=False**)

clf**.**fit(digits**.**data, digits**.**target)

scores**.**append({

'model': model\_name,

'best\_score': clf**.**best\_score\_,

'best\_params': clf**.**best\_params\_

})

df **=** pd**.**DataFrame(scores,columns**=**['model','best\_score','best\_params'])

df

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model\_selection\\_search.py:841: DeprecationWarning: The default of the `iid` parameter will change from True to False in version 0.22 and will be removed in 0.24. This will change numeric results when test-set sizes are unequal.

DeprecationWarning)

Out[4]:

|  | **model** | **best\_score** | **best\_params** |
| --- | --- | --- | --- |
| **0** | svm | 0.949360 | {'C': 1, 'kernel': 'linear'} |
| **1** | random\_forest | 0.899833 | {'n\_estimators': 10} |
| **2** | logistic\_regression | 0.920979 | {'C': 1} |
| **3** | naive\_bayes\_gaussian | 0.806344 | {} |
| **4** | naive\_bayes\_multinomial | 0.871452 | {} |
| **5** | decision\_tree | 0.817474 | {'criterion': 'entropy'} |

**For me the winner is svm (C=1, kernel=linear) with 94.93% score. It could be different for you as I have limited my parameters to be certain values only**