Emma Mayes (eemayes2)

IE517 MLF F21

Module 7 Homework (Random Forest)

Using the ccdefault dataset, and 10 fold cross validation described in Raschka;

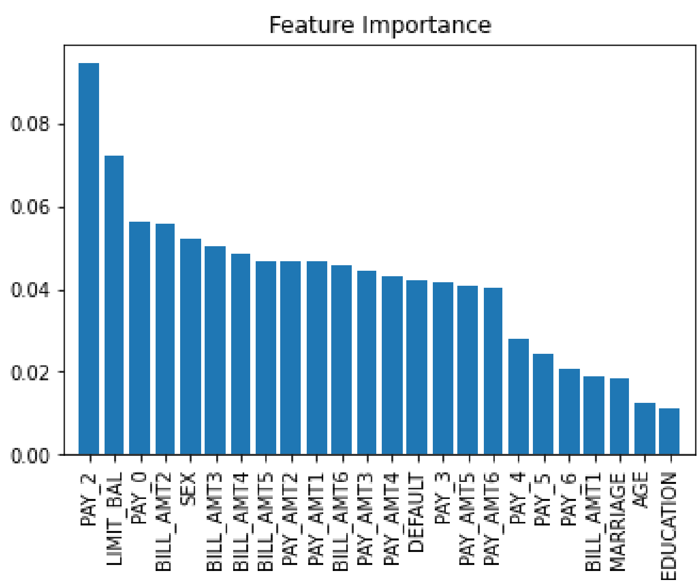
Part 1: Random forest estimators

Fit a random forest model, try several different values for N_estimators, report in-sample accuracies.

I tried n_estimators = 10, 20, 50, 75, 100, 150, 200, 400

Part 2: Random forest feature importance

Display the individual feature importance of your best model in Part 1 above using the code presented in Chapter 4 on page 136. {importances=forest.feature_importances_ }



Part 3: Conclusions

Write a short paragraph summarizing your findings. Answer the following questions:

a) What is the relationship between n_estimators, in-sample CV accuracy and computation time?
The greater the number of estimators, the greater the computation time, but also the greater the in-sample accuracy
b) What is the optimal number of estimators for your forest?
N_estimators = 75. Out of the range of n_estimators I ran, this gave the best in-sample and out-of-sample accuracy scores partnered with the shorted computation time, as more estimators were able to perform similarly, just with a longer time to train.
c) Which features contribute the most importance in your model according to scikit-learn function?

Pay_2 did, with it given 0.094387 as its feature importance

d) What is feature importance and how is it calculated? (If you are not sure, refer to the Scikit-Learn.org documentation.)
Feature importance is the mean and standard deviation based on how much decrease of impurity a given feature provides within each tree in the random forest

Part 4: Appendix

Link to github repo: https://github.com/eemayes2/IE517_F21_HWK7

IE517_HWK7

October 8, 2021

[1]:
```
#Import libraries needed
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')


#Check if any null values we need to change
def num_missing(x):
    return sum(x.isnull())
```

[2]:
```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount, call
drive.mount("/content/gdrive", force_remount=True).
```

[3]:
```
#Read in Data
df = pd.read_csv('gdrive/MyDrive/Colab Notebooks/ccdefault.csv', header=0)
df.head()
```

[3]:
```
   ID  LIMIT_BAL  SEX  EDUCATION  ...  PAY_AMT4  PAY_AMT5  PAY_AMT6  DEFAULT
0   1      20000    2          2  ...         0         0         0        1
1   2     120000    2          2  ...      1000         0      2000        1
2   3      90000    2          2  ...      1000      1000      5000        0
3   4      50000    2          2  ...      1100      1069      1000        0
4   5      50000    1          2  ...      9000       689       679        0

[5 rows x 25 columns]
```

[4]:
```
df.describe()
```

[4]:
```
                 ID      LIMIT_BAL  ...      PAY_AMT6       DEFAULT
count  30000.000000   30000.000000  ...  30000.000000  30000.000000
mean   15000.500000  167484.322667  ...   5215.502567      0.221200
std     8660.398374  129747.661567  ...  17777.465775      0.415062
min        1.000000   10000.000000  ...      0.000000      0.000000
```

```
25%      7500.750000     50000.000000  ...     117.750000       0.000000
50%     15000.500000    140000.000000  ...    1500.000000       0.000000
75%     22500.250000    240000.000000  ...    4000.000000       0.000000
max     30000.000000   1000000.000000  ...  528666.000000       1.000000

[8 rows x 25 columns]
```

0.1 Ten-fold Cross-Validation

[5]:
```python
from sklearn.model_selection import StratifiedKFold, cross_val_score
x = df.drop(columns = ['DEFAULT'])
y = df['DEFAULT']

kf = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)
```

0.2 Random Forest from Raschka Ch. 3

[6]:
```python
from sklearn.ensemble import RandomForestClassifier
#Grid Search takes too long, so just do it in a nested loop
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score
```

[11]:
```python
kf.get_n_splits(x,y)
i = 1
parameters = [10, 20, 50, 75, 100, 150, 200, 400]
out_of = []
in_of = []
acc_in = []
acc_out = []
in_mean = []

for param in parameters:
  forest = RandomForestClassifier(criterion = 'gini', n_estimators = param)
  print("For n_estimators = " + str(param))
  for train_index, test_index in kf.split(x,y):
    X_train, X_test = x.iloc[list(train_index)], x.iloc[list(test_index)]
    y_train, y_test = y.iloc[list(train_index)], y.iloc[list(test_index)]
    forest.fit(X_train, y_train)
    y_pred = forest.predict(X_test)
    acc_out = accuracy_score(y_test, y_pred)
    out_of.append(acc_out)
    y_pred_train = forest.predict(X_train)
    acc_in = accuracy_score(y_train, y_pred_train)
    in_of.append(acc_in)
    print("\tFold Index:" + str(i) + "\n\t\tOut of Sample Accuracy: " +␣
 ↪str(acc_out))
    print("\t\tIn Sample Accuracy: " + str(acc_in))
```

```
        i += 1
    in_mean.append(np.mean(acc_in))
    print("Mean In-Sample Accuracy for " + str(param) + " n_estimators: " +␣
 ↪str(np.mean(acc_in)))
    i = 1
```

```
For n_estimators = 10
        Fold Index:1
                Out of Sample Accuracy: 0.7993333333333333
                In Sample Accuracy: 0.9801111111111112
        Fold Index:2
                Out of Sample Accuracy: 0.8216666666666667
                In Sample Accuracy: 0.9805925925925926
        Fold Index:3
                Out of Sample Accuracy: 0.805
                In Sample Accuracy: 0.9805185185185186
        Fold Index:4
                Out of Sample Accuracy: 0.816
                In Sample Accuracy: 0.980925925925926
        Fold Index:5
                Out of Sample Accuracy: 0.802
                In Sample Accuracy: 0.9794074074074074
        Fold Index:6
                Out of Sample Accuracy: 0.8116666666666666
                In Sample Accuracy: 0.9803333333333333
        Fold Index:7
                Out of Sample Accuracy: 0.797
                In Sample Accuracy: 0.9818888888888889
        Fold Index:8
                Out of Sample Accuracy: 0.809
                In Sample Accuracy: 0.9798888888888889
        Fold Index:9
                Out of Sample Accuracy: 0.8046666666666666
                In Sample Accuracy: 0.9800740740740741
        Fold Index:10
                Out of Sample Accuracy: 0.8053333333333333
                In Sample Accuracy: 0.9821851851851852
Mean In-Sample Accuracy for 10 n_estimators: 0.9821851851851852
For n_estimators = 20
        Fold Index:1
                Out of Sample Accuracy: 0.807
                In Sample Accuracy: 0.9938518518518519
        Fold Index:2
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 0.9936296296296296
        Fold Index:3
                Out of Sample Accuracy: 0.8133333333333334
```

```
                In Sample Accuracy: 0.9935925925925926
        Fold Index:4
                Out of Sample Accuracy: 0.8193333333333334
                In Sample Accuracy: 0.9930740740740741
        Fold Index:5
                Out of Sample Accuracy: 0.8093333333333333
                In Sample Accuracy: 0.9945555555555555
        Fold Index:6
                Out of Sample Accuracy: 0.8153333333333334
                In Sample Accuracy: 0.9938518518518519
        Fold Index:7
                Out of Sample Accuracy: 0.803
                In Sample Accuracy: 0.9935555555555555
        Fold Index:8
                Out of Sample Accuracy: 0.8083333333333333
                In Sample Accuracy: 0.9941481481481481
        Fold Index:9
                Out of Sample Accuracy: 0.8133333333333334
                In Sample Accuracy: 0.9936296296296296
        Fold Index:10
                Out of Sample Accuracy: 0.822
                In Sample Accuracy: 0.9932222222222222
Mean In-Sample Accuracy for 20 n_estimators: 0.9932222222222222
For n_estimators = 50
        Fold Index:1
                Out of Sample Accuracy: 0.8136666666666666
                In Sample Accuracy: 0.9994444444444445
        Fold Index:2
                Out of Sample Accuracy: 0.8216666666666667
                In Sample Accuracy: 0.9995185185185185
        Fold Index:3
                Out of Sample Accuracy: 0.8146666666666667
                In Sample Accuracy: 0.9994444444444445
        Fold Index:4
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 0.9991851851851852
        Fold Index:5
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 0.9995555555555555
        Fold Index:6
                Out of Sample Accuracy: 0.8196666666666667
                In Sample Accuracy: 0.9993703703703704
        Fold Index:7
                Out of Sample Accuracy: 0.808
                In Sample Accuracy: 0.9993703703703704
        Fold Index:8
                Out of Sample Accuracy: 0.8183333333333334
                In Sample Accuracy: 0.9994814814814815
```

```
        Fold Index:9
                Out of Sample Accuracy: 0.8173333333333334
                In Sample Accuracy: 0.9993333333333333
        Fold Index:10
                Out of Sample Accuracy: 0.82
                In Sample Accuracy: 0.9994074074074074
Mean In-Sample Accuracy for 50 n_estimators: 0.9994074074074074
For n_estimators = 75
        Fold Index:1
                Out of Sample Accuracy: 0.8143333333333334
                In Sample Accuracy: 0.9998888888888889
        Fold Index:2
                Out of Sample Accuracy: 0.8263333333333334
                In Sample Accuracy: 0.9999629629629629
        Fold Index:3
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 0.9998888888888889
        Fold Index:4
                Out of Sample Accuracy: 0.8233333333333334
                In Sample Accuracy: 0.9999259259259259
        Fold Index:5
                Out of Sample Accuracy: 0.8106666666666666
                In Sample Accuracy: 0.9999259259259259
        Fold Index:6
                Out of Sample Accuracy: 0.8166666666666667
                In Sample Accuracy: 0.9999629629629629
        Fold Index:7
                Out of Sample Accuracy: 0.8053333333333333
                In Sample Accuracy: 0.9999629629629629
        Fold Index:8
                Out of Sample Accuracy: 0.818
                In Sample Accuracy: 0.9998518518518519
        Fold Index:9
                Out of Sample Accuracy: 0.8093333333333333
                In Sample Accuracy: 1.0
        Fold Index:10
                Out of Sample Accuracy: 0.8213333333333334
                In Sample Accuracy: 0.9999629629629629
Mean In-Sample Accuracy for 75 n_estimators: 0.9999629629629629
For n_estimators = 100
        Fold Index:1
                Out of Sample Accuracy: 0.8173333333333334
                In Sample Accuracy: 1.0
        Fold Index:2
                Out of Sample Accuracy: 0.8293333333333334
                In Sample Accuracy: 1.0
        Fold Index:3
                Out of Sample Accuracy: 0.815
```

```
                In Sample Accuracy: 1.0
        Fold Index:4
                Out of Sample Accuracy: 0.8193333333333334
                In Sample Accuracy: 1.0
        Fold Index:5
                Out of Sample Accuracy: 0.817
                In Sample Accuracy: 1.0
        Fold Index:6
                Out of Sample Accuracy: 0.8206666666666667
                In Sample Accuracy: 1.0
        Fold Index:7
                Out of Sample Accuracy: 0.8066666666666666
                In Sample Accuracy: 0.9999629629629629
        Fold Index:8
                Out of Sample Accuracy: 0.822
                In Sample Accuracy: 1.0
        Fold Index:9
                Out of Sample Accuracy: 0.8123333333333334
                In Sample Accuracy: 0.9999259259259259
        Fold Index:10
                Out of Sample Accuracy: 0.824
                In Sample Accuracy: 1.0
Mean In-Sample Accuracy for 100 n_estimators: 1.0
For n_estimators = 150
        Fold Index:1
                Out of Sample Accuracy: 0.814
                In Sample Accuracy: 1.0
        Fold Index:2
                Out of Sample Accuracy: 0.826
                In Sample Accuracy: 1.0
        Fold Index:3
                Out of Sample Accuracy: 0.8183333333333334
                In Sample Accuracy: 1.0
        Fold Index:4
                Out of Sample Accuracy: 0.822
                In Sample Accuracy: 1.0
        Fold Index:5
                Out of Sample Accuracy: 0.8146666666666667
                In Sample Accuracy: 0.9999629629629629
        Fold Index:6
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 0.9999629629629629
        Fold Index:7
                Out of Sample Accuracy: 0.804
                In Sample Accuracy: 1.0
        Fold Index:8
                Out of Sample Accuracy: 0.8203333333333334
                In Sample Accuracy: 1.0
```

```
        Fold Index:9
                Out of Sample Accuracy: 0.8146666666666667
                In Sample Accuracy: 1.0
        Fold Index:10
                Out of Sample Accuracy: 0.8283333333333334
                In Sample Accuracy: 1.0
Mean In-Sample Accuracy for 150 n_estimators: 1.0
For n_estimators = 200
        Fold Index:1
                Out of Sample Accuracy: 0.818
                In Sample Accuracy: 1.0
        Fold Index:2
                Out of Sample Accuracy: 0.8293333333333334
                In Sample Accuracy: 1.0
        Fold Index:3
                Out of Sample Accuracy: 0.8163333333333334
                In Sample Accuracy: 1.0
        Fold Index:4
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 1.0
        Fold Index:5
                Out of Sample Accuracy: 0.818
                In Sample Accuracy: 1.0
        Fold Index:6
                Out of Sample Accuracy: 0.8216666666666667
                In Sample Accuracy: 1.0
        Fold Index:7
                Out of Sample Accuracy: 0.809
                In Sample Accuracy: 1.0
        Fold Index:8
                Out of Sample Accuracy: 0.8186666666666667
                In Sample Accuracy: 1.0
        Fold Index:9
                Out of Sample Accuracy: 0.814
                In Sample Accuracy: 1.0
        Fold Index:10
                Out of Sample Accuracy: 0.826
                In Sample Accuracy: 1.0
Mean In-Sample Accuracy for 200 n_estimators: 1.0
For n_estimators = 400
        Fold Index:1
                Out of Sample Accuracy: 0.816
                In Sample Accuracy: 1.0
        Fold Index:2
                Out of Sample Accuracy: 0.8286666666666667
                In Sample Accuracy: 1.0
        Fold Index:3
                Out of Sample Accuracy: 0.814
```

```
            In Sample Accuracy: 1.0
    Fold Index:4
            Out of Sample Accuracy: 0.8216666666666667
            In Sample Accuracy: 1.0
    Fold Index:5
            Out of Sample Accuracy: 0.814
            In Sample Accuracy: 1.0
    Fold Index:6
            Out of Sample Accuracy: 0.8196666666666667
            In Sample Accuracy: 1.0
    Fold Index:7
            Out of Sample Accuracy: 0.8036666666666666
            In Sample Accuracy: 1.0
    Fold Index:8
            Out of Sample Accuracy: 0.8186666666666667
            In Sample Accuracy: 1.0
    Fold Index:9
            Out of Sample Accuracy: 0.8173333333333334
            In Sample Accuracy: 1.0
    Fold Index:10
            Out of Sample Accuracy: 0.8236666666666667
            In Sample Accuracy: 1.0
Mean In-Sample Accuracy for 400 n_estimators: 1.0
```

[12]:
```
#Print in-sample accuracies for each model type
print(in_mean)
```

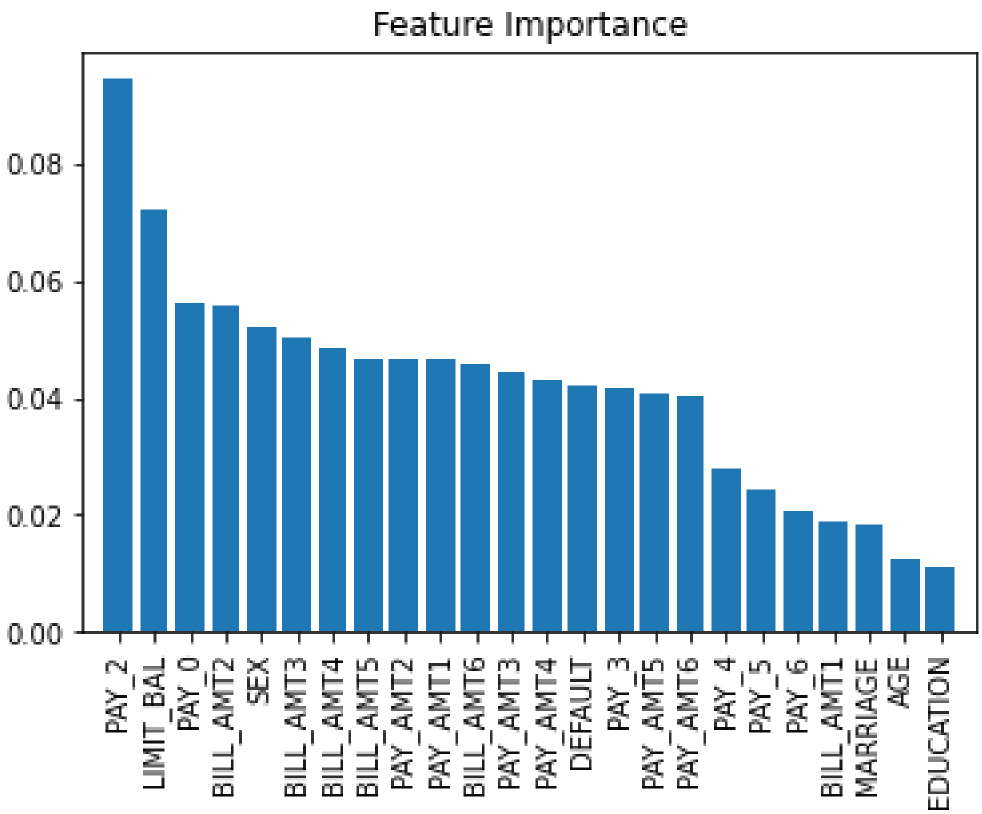
```
[0.9821851851851852, 0.9932222222222222, 0.9994074074074074, 0.9999629629629629,
1.0, 1.0, 1.0, 1.0]
```

[17]:
```
forest_best = RandomForestClassifier(n_estimators = 75, random_state = 1)
forest_best.fit(X_train, y_train)
importances = forest_best.feature_importances_
print(importances)
indices = np.argsort(importances)[: :-1]
feat_labels = df.columns[1:]

for f in range(X_train.shape[1]):
  print("%2d) %-*s %f" % (f + 1, 30, feat_labels[indices[f]],␣
 ↪importances[indices[f]]))

plt.title('Feature Importance')
plt.bar(range(X_train.shape[1]), importances[indices], align = 'center')
plt.xticks(range(X_train.shape[1]), feat_labels[indices], rotation = 90)
plt.xlim([-1, X_train.shape[1]])
```

```
[0.0723393  0.05204395 0.01085784 0.01853991 0.0122156  0.05601438
 0.09438701 0.04177386 0.02788596 0.02409151 0.02087141 0.01895116
```

```
 0.05592564 0.05026756 0.04853628 0.04661175 0.04554953 0.04644812
 0.0465206  0.04448248 0.0428961  0.0405147  0.04030299 0.04197236]
 1) PAY_2                          0.094387
 2) LIMIT_BAL                      0.072339
 3) PAY_0                          0.056014
 4) BILL_AMT2                      0.055926
 5) SEX                            0.052044
 6) BILL_AMT3                      0.050268
 7) BILL_AMT4                      0.048536
 8) BILL_AMT5                      0.046612
 9) PAY_AMT2                       0.046521
10) PAY_AMT1                       0.046448
11) BILL_AMT6                      0.045550
12) PAY_AMT3                       0.044482
13) PAY_AMT4                       0.042896
14) DEFAULT                        0.041972
15) PAY_3                          0.041774
16) PAY_AMT5                       0.040515
17) PAY_AMT6                       0.040303
18) PAY_4                          0.027886
19) PAY_5                          0.024092
20) PAY_6                          0.020871
21) BILL_AMT1                      0.018951
22) MARRIAGE                       0.018540
23) AGE                            0.012216
24) EDUCATION                      0.010858
```

[17]: (-1.0, 24.0)



[18]:
```
print("My name is Emma Mayes")
print("My NetID is: eemayes2")
print("I hereby certify that I have read the University policy on Academic␣
 ↪Integrity and that I am not in violation.")
```

```
My name is Emma Mayes
My NetID is: eemayes2
I hereby certify that I have read the University policy on Academic Integrity
and that I am not in violation.
```

[ ]:
```
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('IE517_HWK7.ipynb')
```

```
--2021-10-08 22:24:12--  https://raw.githubusercontent.com/brpy/colab-
pdf/master/colab_pdf.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)...
185.199.111.133, 185.199.110.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
```

```
Length: 1864 (1.8K) [text/plain]
Saving to: colab_pdf.py

colab_pdf.py        100%[===================>]   1.82K  --.-KB/s    in 0s

2021-10-08 22:24:12 (36.6 MB/s) - colab_pdf.py saved [1864/1864]

Mounted at /content/drive/

WARNING: apt does not have a stable CLI interface. Use with caution in scripts.


WARNING: apt does not have a stable CLI interface. Use with caution in scripts.

Extracting templates from packages: 100%
```