Προ - επεξεργασία και Κατηγοριοποίηση με Python

Καλογερόπουλος Νικήτας - Ρήγας
Μπομπότας Αγοράκης

Τμήμα Μηχανικών Η/Υ & Πληροφορικής
Πανεπιστήμιο Πατρών

25 Μαρτίου 2023

Pandas

Pandas

Pandas

Ένα γρήγορο ισχυρό ευέλικτο και εύχρηστο εργαλείο για διαχείριση και ανάλυση δεδομένων βασισμένο στην γλώσσα προγραμματισμού Python.

- Ποικιλομορφία στο είδος των αρχείων που διαχειρίζεται.
- Ποικιλομορφία στο είδος των δεδομένων που διαχειρίζεται (κείμενα, χρονοσειρές, αριθμούς κ.α.).
- Ευκολία σε διαχείριση (διαχωρισμό, δεικτοδότηση, ένωση και αναδιαμόρφωση).
- Υπολογισμός στατιστικών μεγεθών.
- Οπτικοποίηση δεδομένων (Matplotlib και Seaborn).
- Αποθήκευση δεδομένων είτε σε αρχεία είτε σε κάποια βάση δεδομένων.

Χρήση και Εγκατάσταση

Εγκατάσταση:

- pip install pandas.
- conda install pandas
- Με χρήση κάποιου package manager κάποιου εργαλείου ανάπτυξης (IDE), π.χ Jetbrains PyCharm.

Χρήση:

- import pandas as pd
- from pandas import (function/module)

Series και DataFrames

- Αποτελούν τα δύο βασικά δομικά παρεχόμενα στοιχεία.
- Series: Ένα διάνυσμα που περιέχει στοιχεία ίδιου τύπου. Τα στοιχεία μπορούν να αλλάξουν (mutable), όμως το μέγεθος του διανύσματος δεν μπορεί. Ουσιαστικά αποτελέι μία στήλη.
- DataFrames: Μία δισδιάστατη δομή που αποτελείται από μία συλλογή Series.Τόσο το περιεχόμενο όσο και το μέγεθος μπορεί να μεταβληθεί.

Κατασκευή και Αποθήκευση DataFrame

Κατασκευή:

1. Λεξικό (Dictionary):
   - df = pd.DataFrame(data in dictionary)
2. CSV:
   - df = pd.read_csv('data_in_csv.csv')
3. JSON:
   - df = pd.read_json('data_in_json.json)
4. SQL:
   - df = pd.read_sql_query('select * from table', con)

Αποθήκευση:

1. CSV:
   - df.to_csv('name_file.csv')
2. JSON:
   - df.to_json('name_file.json)
3. SQL:
   - df.to_sql('table_name', con)

Πράξεις σε DataFrames - Προβολή

- Προβολή δεδομένων:
  - df.head() $\longrightarrow$ εμφανίζει τα πρώτα στοιχεία.
  - df.tail() $\longrightarrow$ εμφανίζει τα τελευταία στοιχεία.
  - Με τη χρήση κάποιου ακεραίου ορίσματος καθορίζεται ο αριθμός των στοιχείων προς προβολή, αλλιώς αυτόματα θα είναι 5.
- Πληροφορία για τα δεδομένα:
  - df.info() $\longrightarrow$ εμφανίζει ονόματα στηλών και για κάθε στήλη αριθμό έγκυρων στοιχείων και είδος δεδομένων.
  - Η συνάρτηση describe() πάνω σε κάποια στήλη ή DataFrame $\longrightarrow$ στατιστικά στοιχεία για τη στήλη/DataFrame.

Πράξεις σε DataFrames - Διαχείριση Στηλών

1. Ένωση
   - df.append(another_df)
2. Διαγραφή
   - df.drop(rows_or_columns,axis =1)
   - df.drop_duplicates(inplace = True)
3. Επιλογή στήλης
   - col = df['name'] $\longrightarrow$ Series
   - col = df[['name']] $\longrightarrow$ DataFrame
   - col = df[['name','Surname']] $\longrightarrow$ DataFrame
4. Επιλογή γραμμής
   - row = df.loc['a record']
   - row = df.iloc[index]

Πράξεις σε DataFrames - Επιλογή Υποσυνόλου

Επιλογή υποσυνόλου όπου ικανοποιείται κάποια συνθήκη:

- condition = (movies_df['director'] == "Ridley Scott")
- movies_df[movies_df['director'] == "Ridley Scott"]
- movies_df[movies_df['rating'] >= 8.6]
- movies_df[movies_df['director'].isin(['Christopher Nolan', 'Ridley Scott'])]
- movies_df[((movies_df['year'] >= 2005) & (movies_df['year'] <= 2010)) & (movies_df['rating'] > 8.0) & (movies_df['revenue_millions'] < movies_df['revenue_millions'].quantile(0.25))]

Πράξεις σε DataFrames

Χειρισμός NaN τιμών:

1. df.isnull()
2. df.isnull().sum()
3. df.dropna()
4. df.dropna(axis =1)

Στατιστικά μεγέθη:

1. df['column'].value_counts()
2. df['column'].sum()
3. df['column'].mean()
4. df['column'].median()
5. df.corr()

scikit - learn

scikit - learn

scikit - learn

Scikit-learn (γνωστό και ως sklearn) είναι μια δημοφιλής ανοιχτού κώδικα βιβλιοθήκη μηχανικής μάθησης για τη γλώσσα προγραμματισμού Python. Είναι χτισμένο πάνω στο NumPy, SciPy και matplotlib.

Προσφέρει:

- Αλγόριθμους επιβλεπόμενης και μη επιβλεπόμενης μάθησης.
- Εργαλεία προ επεξεργασίας δεδομένων.
- Εργαλεία επιλογής και αξιολόγησης μοντέλων.
- Εύκολη πρόσβαση στα δεδομένα και τα εργαλεία της Python και άρα χρήσιμο εργαλείο για την ανάπτυξη μοντέλων μηχανικής μάθησης.
- Επεκτασιμότητα και ευελιξία.
- Εκτεταμένη τεκμηρίωση και κοινότητα χρηστών.

Χρήση και Εγκατάσταση

Εγκατάσταση:

- pip install scikit-learn.
- conda install scikit-learn
- Με χρήση κάποιου package manager κάποιου εργαλείου ανάπτυξης (IDE), π.χ Jetbrains PyCharm.

Χρήση:

- from scikit-learn import (function/module)

** Απαιτεί την ύπαρξη των NumPy και SciPY*

Σύνολα Δεδομένων

1. Διαθέτει ήδη σύνολα δεδομένων:
   - Τα Iris dataset, Wine dataset και Breast cancer dataset για την ταξινόμηση.
   - Τα Boston Housing dataset και Diabetes dataset για την παλινδρόμηση.
   - Τα 20 Newsgroups dataset και Reuters dataset για την επεξεργασία φυσικής γλώσσας.
   - Το Digits dataset για την αναγνώριση προτύπων.
   - Τα California Housing dataset και Forest Fires dataset για την πρόβλεψη.
   - Τα Olivetti Faces dataset, Labeled Faces in the Wild dataset και MNIST dataset για την επεξεργασία εικόνων.
2. Διαχειρίζεται κάθε αριθμητικά δεδομένα αποθηκευμένα σε μορφή μητρώων της NumPy ή αραιών μητρών της SciPy.
3. Άλλους τύπους μητρώων που μπορούν να μετατραπούν εύκολα συνήθως σε DataFrames.

Δομικά Στοιχεία

1. Transformers.
2. Estimators.
3. Pipelines $\longrightarrow$ συνδυάζουν πολλούς transformers και έναν estimator σε ένα αντικείμενο για αυτοματοποίηση της προ επεξεργασίας.

Transformers

Transformers

Οι Transformers είναι αντικείμενα που χρησιμοποιούνται για την προ επεξεργασία δεδομένων πριν από την εκπαίδευση κάποιου μοντέλου. Αυτά τα αντικείμενα μετατρέπουν τα δεδομένα εισόδου σε μια μορφή που είναι πιο κατάλληλη για την διαδικασία.

- Πιο συχνά $\longrightarrow$StandarScaler, OneHotEncoder, CountVectorizer, MinMaxScaler, Imputer....
- Κανονικοποιήση σε εύρος τιμών.
- Μετατροπή κατηγορικών δεδομένων σε αριθμητικά ή διανύσματα αριθμητικών.
- Προ - επεξεργασιά κειμένων.
- Διαχείριση τιμών που απουσιάζουν (NaN τιμές).

Estimators

Estimator

Οι Estimators είναι αντικείμενα που χρησιμοποιούνται για την εκτέλεση αλγορίθμων μηχανικής μάθησης. Αυτά τα αντικείμενα περιέχουν τις παραμέτρους του αλγορίθμου και πρέπει να εκπαιδευτούν σε ένα σύνολο εκπαίδευσης.

- Ουσιαστικά, αποτελούν τους αλγορίθμους μηχανικής μάθησης, που θα χρησιμοποιηθούν.
- Πρέπει να γίνουν *fit* στα δεδομένα.
- Σε περίπτωση προβλήματος πρόβλεψης, αναφερόμαστε σε Predictors.

Εκπαίδευση και Αξιολόγηση Μοντέλων

- Η καλή προσαρμογή στα δεδομένα δεν συνεπάγεται και καλή πρόβλεψη ή κατηγοριοποίηση.
- Διαθέτει μηχανισμούς επαλήθευσης για την εκτίμηση της απόδοσης (k-fold cross validation, Hold out, Leave-one out).
- Μετρικές κατάλληλες για το κάθε πρόβλημα.
  1. Mean Absolute Error, Mean Squared Error, R2 Score.
  2. Accuracy.
  3. Precision, Recall, F1 Score.
  4. Confusion Matrix.

Ανάλυση Περίπτωσης

Titanic - Machine Learning from Disaster

PassengerI	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2.	7.925		S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. James	male		0	0	330877	8.4583		Q
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21.075		S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0	237736	30.0708		C
11	1	3	Sandstrom, Miss. Marguerite Rut	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, Miss. Elizabeth	female	58	0	0	113783	26.55	C103	S
13	0	3	Saundercock, Mr. William Henry	male	20	0	0	A/5. 2151	8.05		S
14	0	3	Andersson, Mr. Anders Johan	male	39	1	5	347082	31.275		S
15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14	0	0	350406	7.8542		S
16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55	0	0	248706	16		S
17	0	3	Rice, Master. Eugene	male	2	4	1	382652	29.125		Q
18	1	2	Williams, Mr. Charles Eugene	male		0	0	244373	13		S
19	0	3	Vander Planke, Mrs. Julius (Emelia Maria Vandemoortele)	female	31	1	0	345763	18		S
20	1	3	Masselmani, Mrs. Fatima	female		0	0	2649	7.225		C
21	0	2	Fynney, Mr. Joseph J	male	35	0	0	239865	26		S
22	1	2	Beesley, Mr. Lawrence	male	34	0	0	248698	13	D56	S
23	1	3	McGowan, Miss. Anna "Annie"	female	15	0	0	330923	8.0292		Q

Titanic - Machine Learning from Disaster

- **survival**: Επιβίωση 0 = όχι, 1 = ναι.
- pclass:κατηγορία εισιτήριου (1, 2, 3).
- sex: φύλο.
- Age: ηλικία σε χρόνια.
- sibsp: # αδελφών ή συζύγων στο πλοίο.
- parch: # γονέων ή παιδιών στο πλοίο.
- ticket: αριθμός εισιτηρίου.
- Fare: κόμιστρο.
- cabin: αριθμός καμπίνας.
- embarked: Λιμάνι επιβίβασης.

Εισαγωγή δεδομένων

```
In 186  1  # data visualization
        2  import seaborn as sns
        3  %matplotlib inline
        4  from matplotlib import pyplot as plt
        5  import pandas as pd
        6
        7
        8  test_set = "titanic/test.csv"
        9  train_set = "titanic/train.csv"
       10  test_df = pd.read_csv(test_set)
       11  train_df = pd.read_csv(train_set)
       12  #test_df.info()
       13  train_df.info()
       14  train_df.describe()
```

Σχήμα: Εισαγωγή Titanic Dataset

Στατιστικά για toDataFrame

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Σχήμα: Titanic DataFrame

Στατιστικά για το Dataset

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Σχήμα: Titanic Dataset - Στατιστικά

Εντοπισμός Τιμών που Απουσιάζουν

```
NaNs_data_df =  train_df.isnull().sum().sort_values(ascending=False)
print(NaNs_data_df)
```

```
Cabin          687
Age            177
Embarked         2
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
SibSp            0
Parch            0
Ticket           0
```

Σχήμα: NaNs - Στατιστικά

Χειρισμός Τιμών που Απουσιάζουν

```python
train_df["Age"] = train_df["Age"].fillna(train_df["Age"].mean())
train_df["Cabin"] = train_df["Cabin"].fillna('U')
train_df["Embarked"] = train_df["Embarked"].fillna(train_df["Embarked"].mode()[0])

NaNs_data_df =  train_df.isnull().sum().sort_values(ascending=False)
print(NaNs_data_df)
train_df.head(20)
```

```
PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age            0
SibSp          0
Parch          0
Ticket         0
Fare           0
Cabin          0
```

Σχήμα: Missing Values

Κατηγορικά Δεδομένα - Φύλο

```
train_df['Sex'] = train_df['Sex'].map({'female':0,'male':1})
print(train_df.head())
```

```
   1           2         1       1
   2           3         1       3
   3           4         1       1
   4           5         0       3

                                                 Name  Sex   Age  SibSp  Parch
0                             Braund, Mr. Owen Harris    1  22.0      1      0
1  Cumings, Mrs. John Bradley (Florence Briggs Th...    0  38.0      1      0
2                              Heikkinen, Miss. Laina    0  26.0      0      0
3        Futrelle, Mrs. Jacques Heath (Lily May Peel)    0  35.0      1      0
4                            Allen, Mr. William Henry    1  35.0      0      0
```

Σχήμα: Χειρισμός Φύλου

Κατηγορικά Δεδομένα - Λιμάνι Επιβίβασης

```
print(train_df['Embarked'].describe())
train_df = pd.get_dummies(train_df,columns=["Embarked"])
print(train_df.head())
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name  Sex   Age  SibSp  Parch  \
0                            Braund, Mr. Owen Harris    1  22.0      1      0
1  Cumings, Mrs. John Bradley (Florence Briggs Th...    0  38.0      1      0
2                             Heikkinen, Miss. Laina    0  26.0      0      0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)    0  35.0      1      0
4                           Allen, Mr. William Henry    1  35.0      0      0

             Ticket     Fare Cabin  Embarked_C  Embarked_Q  Embarked_S
0         A/5 21171   7.2500     U           0           0           1
1          PC 17599  71.2833   C85           1           0           0
2  STON/O2. 3101282   7.9250     U           0           0           1
3            113803  53.1000  C123           0           0           1
4            373450   8.0500     U           0           0           1
```

Κατηγορικά Δεδομένα - Καμπίνα

```python
print(train_df['Cabin'])

deck_set = train_df["Cabin"].str[0]
print(deck_set)

train_df['Cabin'] = deck_set
print(train_df['Cabin'].describe())

unique_list = list(train_df.Cabin.unique())
mapping = {item: unique_list.index(item) for item in unique_list}
train_df['Cabin'] = train_df['Cabin'].map(mapping)
print(train_df.head(10))
```

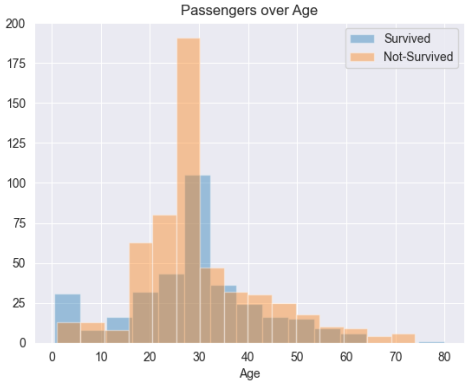
```
   Parch            Ticket     Fare  Cabin  Embarked_C  Embarked_Q  Embarked_S
0      0         A/5 21171   7.2500      0           0           0           1
1      0          PC 17599  71.2833      1           1           0           0
2      0  STON/O2. 3101282   7.9250      0           0           0           1
3      0            113803  53.1000      1           0           0           1
4      0            373450   8.0500      0           0           0           1
5      0            330877   8.4583      0           0           1           0
6      0             17463  51.8625      2           0           0           1
7      1            349909  21.0750      0           0           0           1
8      2            347742  11.1333      0           0           0           1
```

Συσχετίσεις: Ηλικία - Επιβίωση

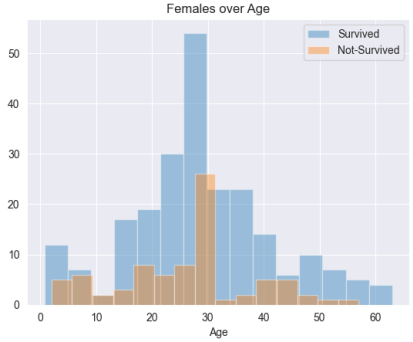
```
plt.figure('Passengers over Age')
sns.distplot(train_df[train_df['Survived']==1].Age.dropna(),label="Survived",bins=15,kde =False)
sns.distplot(train_df[train_df['Survived']==0].Age.dropna(),label="Not-Survived",bins=15,kde = False)
plt.legend()
plt.title(label='Passengers over Age')
plt.show()
```

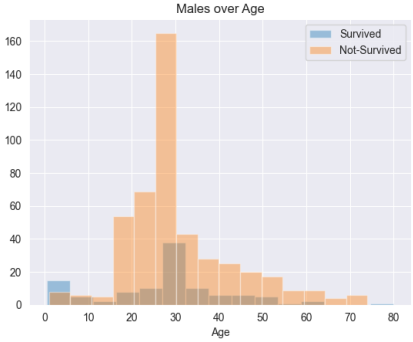
Σχήμα: Python - Σχεδιασμός Κατανομής

Συσχετίσεις: Ηλικία - Επιβίωση

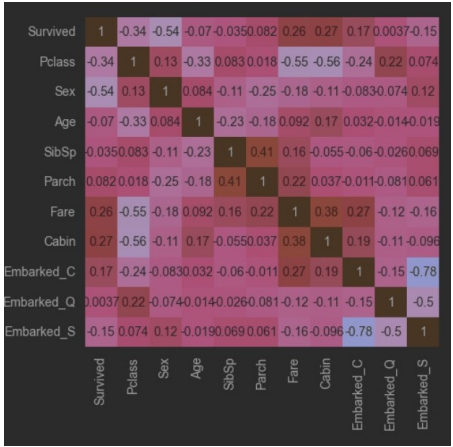


Συσχετίσεις: Φύλο - Ηλικία - Επιβίωση





Διάγραμμα Συσχετίσεων



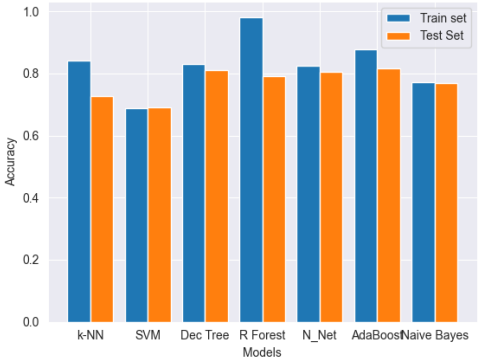
Επιλογή Κατηγοριοποιητών

```
def select_classifier(option):
    if option == 1:
        classifier = KNeighborsClassifier(p=1, n_neighbors=3)
    elif option == 2:
        classifier = SVC(kernel="linear", C=0.001, random_state=42)
    elif option == 3:
        classifier = DecisionTreeClassifier(max_depth=3, criterion='entropy', random_state=42)
    elif option == 4:
        classifier = RandomForestClassifier(n_estimators=100, criterion='entropy', random_state=42)
    elif option == 5:
        classifier = MLPClassifier(max_iter=10000, activation='logistic', random_state=42)
    elif option == 6:
        classifier = AdaBoostClassifier(n_estimators=200, random_state=42)
    elif option == 7:
        classifier = GaussianNB()
    else:
        raise ValueError("option values are : 1,2,3,4,5,6,7,8")
    return classifier
```

Κατηγοριοποίηση

```
def run_classification(X,Y,classifier = 1):
    X_train, X_test, y_train, y_test = train_test_split( X, Y, test_size=0.3, random_state=42)
#training
    model = select_classifier(classifier)
    model.fit(X_train,y_train)
    predictions = model.predict(X_test)
    print("------------------------------------------------------------------------")
    print(f"Classifier {option} yeilds training accuracy of {model.score(X_train,y_train)}\n with a testing accuracy of {accuracy_score
     (y_test, predictions)}")
    return classifier,model.score(X_train,y_train),accuracy_score(y_test, predictions)
```

Αποτελέσματα Κατηγοριοποίησης



Βελτίωση Παραμέτρων - GridSearch

```
%%time

from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

estimators = [i for i in range(10,210,10)]
depth = [i for i in range(1,24,4)]
params = {"n_estimators":estimators,
          "max_depth":depth}
X_train, X_test, y_train, y_test = train_test_split( X, Y, test_size=0.3, random_state=42)
random_forest = RandomForestClassifier(criterion='entropy', random_state=42)
random_forest = GridSearchCV(random_forest,params,scoring='accuracy',n_jobs=4)
random_forest.fit(X_train,y_train)

print(f"Best Parameters: {random_forest.best_params_}")
print(f"Best mean cv accuracy: {random_forest.best_score_}")

y = random_forest.best_estimator_.predict(X_test)
print(f'Accuracy after GridSearch: {accuracy_score(y_test, y)}')
```

```
Best Parameters: {'max_depth': 5, 'n_estimators': 180}
Best mean cv accuracy: 0.8346580645161291
Accuracy after GridSearch: 0.8134328358208955
CPU times: total: 375 ms
Wall time: 31.2 s
```

Σχολιασμός Ακρίβειας

- Πολλαπλά Μοντέλα:
  1. Βέλτιστη απόδοση AdaBoost
  2. Overfitting σε πολλά μοντέλα.
  3. Random Forest: ακραίο Overfitting.
  4. Παρόμοια απόδοση σε αρκετά μοντέλα.
- GridSearch:
  1. Επιλογή και καθορισμός παραμέτρων
  2. Ορισμός μοντέλου προς βελτιστοποίηση
  3. Εφαρμογή παραμέτρων στο μοντέλο που επιλέχθηκε $\longrightarrow$ GridSearchcv
  4. Εφαρμογή καλύτερου εκτιμητή στο σύνολο ελέγχου.

** Δοκιμάστε να βελτιώσετε τα αποτελέσματα στο Jupyter NoteBook που συνοδεύεται.*

Παρατηρήσεις

Για την βελτίωση των αποτελεσμάτων:

1. Έλεγχος αναπαραστάσεων ως προς την εκφραστικότητα και την επιρροή τους στον εκάστοτε αλγόριθμο.
2. Περισσότεροι μετασχηματισμοί χαρακτηριστικών:
   - Μετατροπή της ηλικίας σε εύρος ηλικιών - κλάσεις.
   - Τα ονόματα περιλαμβάνουν τίτλους.
   - Χαρακτηριστικό που να εκφράζει την οικονομική επιφάνεια (συνδυασμός τίτλου και κομίστρου).
   - Ύπαρξή ομάδας φιλικά προσκείμενης.
3. Έλεγχος σημαντικότητας χαρακτηριστικών.
4. Παραμετροποίηση.
5. Συνδυαστικά μοντέλα.