Final_project

June 1, 2022

Classification with Python

In this notebook we try to practice all the classification algorithms that we have learned in this course.

We load a dataset using Pandas library, and apply the following algorithms, and find the best one for this specific dataset by accuracy evaluation methods.

Let's first load required libraries:

[1]:
```
!pip install seaborn
import itertools
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.ticker import NullFormatter
import pandas as pd
import numpy as np
import matplotlib.ticker as ticker
from sklearn import preprocessing
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
import scipy.optimize as opt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn import svm
from sklearn.metrics import jaccard_similarity_score #instead of jaccard_score,␣
 ↪which didn't import
from sklearn.metrics import f1_score
from sklearn.metrics import log_loss
%matplotlib inline
```

```
Requirement already satisfied: seaborn in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (0.9.0)
Requirement already satisfied: scipy>=0.14.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from seaborn)
(1.7.3)
Requirement already satisfied: pandas>=0.15.2 in
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from seaborn)
(1.3.5)
Requirement already satisfied: matplotlib>=1.4.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from seaborn)
(3.5.2)
Requirement already satisfied: numpy>=1.9.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from seaborn)
(1.21.6)
Requirement already satisfied: python-dateutil>=2.7 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (2.8.2)
Requirement already satisfied: packaging>=20.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (21.3)
Requirement already satisfied: cycler>=0.10 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (0.11.0)
Requirement already satisfied: pyparsing>=2.2.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (3.0.9)
Requirement already satisfied: pillow>=6.2.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (8.1.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (1.4.2)
Requirement already satisfied: fonttools>=4.22.0 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
matplotlib>=1.4.3->seaborn) (4.33.3)
Requirement already satisfied: pytz>=2017.3 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
pandas>=0.15.2->seaborn) (2022.1)
Requirement already satisfied: typing-extensions in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from
kiwisolver>=1.0.1->matplotlib>=1.4.3->seaborn) (4.2.0)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.7/site-packages (from python-
dateutil>=2.7->matplotlib>=1.4.3->seaborn) (1.16.0)

/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/utils/validation.py:37: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  LARGE_SPARSE_SUPPORTED = LooseVersion(scipy_version) >= '0.14.0'
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:35: DeprecationWarning: `np.float`
is a deprecated alias for the builtin `float`. To silence this warning, use
`float` by itself. Doing this will not modify any behavior and is safe. If you
specifically wanted the numpy scalar type, use `np.float64` here.
```

```
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=np.finfo(np.float).eps,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:597: DeprecationWarning: `np.float`
is a deprecated alias for the builtin `float`. To silence this warning, use
`float` by itself. Doing this will not modify any behavior and is safe. If you
specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=np.finfo(np.float).eps, copy_X=True, fit_path=True,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:836: DeprecationWarning: `np.float`
is a deprecated alias for the builtin `float`. To silence this warning, use
`float` by itself. Doing this will not modify any behavior and is safe. If you
specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=np.finfo(np.float).eps, copy_X=True, fit_path=True,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:862: DeprecationWarning: `np.float`
is a deprecated alias for the builtin `float`. To silence this warning, use
`float` by itself. Doing this will not modify any behavior and is safe. If you
specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=np.finfo(np.float).eps, positive=False):
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:1097: DeprecationWarning:
`np.float` is a deprecated alias for the builtin `float`. To silence this
warning, use `float` by itself. Doing this will not modify any behavior and is
safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  max_n_alphas=1000, n_jobs=None, eps=np.finfo(np.float).eps,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:1344: DeprecationWarning:
`np.float` is a deprecated alias for the builtin `float`. To silence this
warning, use `float` by itself. Doing this will not modify any behavior and is
safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  max_n_alphas=1000, n_jobs=None, eps=np.finfo(np.float).eps,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/least_angle.py:1480: DeprecationWarning:
`np.float` is a deprecated alias for the builtin `float`. To silence this
warning, use `float` by itself. Doing this will not modify any behavior and is
safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
```

```
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=np.finfo(np.float).eps, copy_X=True, positive=False):
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/randomized_l1.py:152: DeprecationWarning:
`np.float` is a deprecated alias for the builtin `float`. To silence this
warning, use `float` by itself. Doing this will not modify any behavior and is
safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  precompute=False, eps=np.finfo(np.float).eps,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/randomized_l1.py:320: DeprecationWarning:
`np.float` is a deprecated alias for the builtin `float`. To silence this
warning, use `float` by itself. Doing this will not modify any behavior and is
safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=np.finfo(np.float).eps, random_state=None,
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/randomized_l1.py:580: DeprecationWarning:
`np.float` is a deprecated alias for the builtin `float`. To silence this
warning, use `float` by itself. Doing this will not modify any behavior and is
safe. If you specifically wanted the numpy scalar type, use `np.float64` here.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  eps=4 * np.finfo(np.float).eps, n_jobs=None,
```

0.0.1 About dataset

This dataset is about past loans. The **Loan_train.csv** data set includes details of 346 customers whose loan are already paid off or defaulted. It includes following fields:

Field	Description
Loan_status	Whether a loan is paid off on in collection
Principal	Basic principal loan amount at the
Terms	Origination terms which can be weekly (7 days), biweekly, and monthly payoff schedule
Effective_date	When the loan got originated and took effects
Due_date	Since it's one-time payoff schedule, each loan has one single due date
Age	Age of applicant
Education	Education of applicant
Gender	The gender of applicant

Let's download the dataset

[2]:
```
!wget -O loan_train.csv https://cf-courses-data.s3.us.cloud-object-storage.
↪appdomain.cloud/IBMDeveloperSkillsNetwork-ML0101EN-SkillsNetwork/labs/
↪FinalModule_Coursera/data/loan_train.csv
```

```
--2022-05-31 17:54:30--  https://cf-courses-data.s3.us.cloud-object-
storage.appdomain.cloud/IBMDeveloperSkillsNetwork-ML0101EN-
SkillsNetwork/labs/FinalModule_Coursera/data/loan_train.csv
Resolving cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-
courses-data.s3.us.cloud-object-storage.appdomain.cloud)… 169.63.118.104
Connecting to cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud (cf-
courses-data.s3.us.cloud-object-storage.appdomain.cloud)|169.63.118.104|:443…
connected.
HTTP request sent, awaiting response… 200 OK
Length: 23101 (23K) [text/csv]
Saving to: 'loan_train.csv'

loan_train.csv      100%[===================>]  22.56K  95.6KB/s    in 0.2s

2022-05-31 17:54:30 (95.6 KB/s) - 'loan_train.csv' saved [23101/23101]
```

0.0.2 Load Data From CSV File

[3]:
```
df = pd.read_csv('loan_train.csv')
df.head()
```

[3]:

	Unnamed: 0	Unnamed: 0.1	loan_status	Principal	terms	effective_date	due_date	age	education	Gender
0	0	0	PAIDOFF	1000	30	9/8/2016	10/7/2016	45	High School or Below	male
1	2	2	PAIDOFF	1000	30	9/8/2016	10/7/2016	33	Bechalor	female
2	3	3	PAIDOFF	1000	15	9/8/2016	9/22/2016	27	college	male
3	4	4	PAIDOFF	1000	30	9/9/2016	10/8/2016	28	college	female
4	6	6	PAIDOFF	1000	30	9/9/2016	10/8/2016	29	college	male

[4]:
```
df.shape
```

[4]: (346, 10)

0.0.3 Convert to date time object

[5]:
```
df['due_date'] = pd.to_datetime(df['due_date'])
df['effective_date'] = pd.to_datetime(df['effective_date'])
df.head()
```

[5]:
```
   Unnamed: 0  Unnamed: 0.1 loan_status  Principal  terms effective_date  \
0           0             0     PAIDOFF       1000     30     2016-09-08
1           2             2     PAIDOFF       1000     30     2016-09-08
2           3             3     PAIDOFF       1000     15     2016-09-08
3           4             4     PAIDOFF       1000     30     2016-09-09
4           6             6     PAIDOFF       1000     30     2016-09-09

    due_date  age             education  Gender
0 2016-10-07   45  High School or Below    male
1 2016-10-07   33              Bechalor  female
2 2016-09-22   27               college    male
3 2016-10-08   28               college  female
4 2016-10-08   29               college    male
```

1 Data visualization and pre-processing

Let's see how many of each class is in our data set

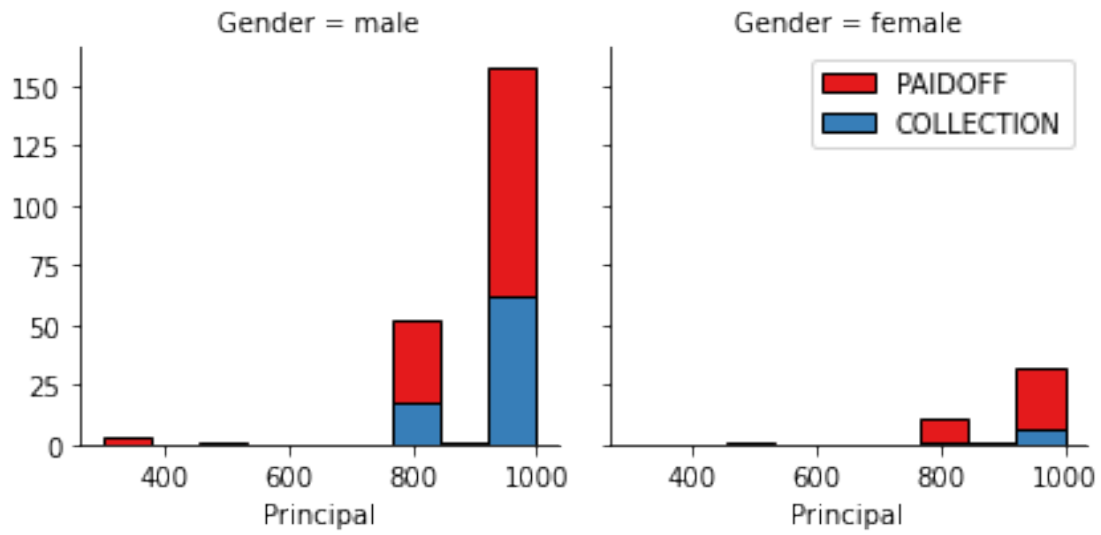
[6]:
```
df['loan_status'].value_counts()
```

[6]:
```
PAIDOFF       260
COLLECTION     86
Name: loan_status, dtype: int64
```

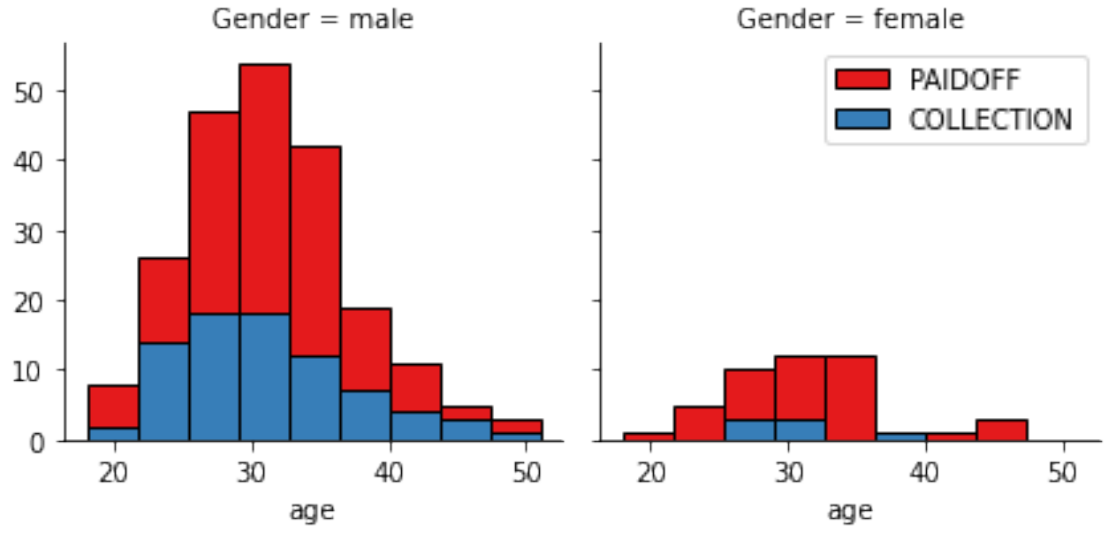
260 people have paid off the loan on time while 86 have gone into collection

Let's plot some columns to underestand data better:

[7]:
```
bins = np.linspace(df.Principal.min(), df.Principal.max(), 10)
g = sns.FacetGrid(df, col="Gender", hue="loan_status", palette="Set1",␣
 ↪col_wrap=2)
g.map(plt.hist, 'Principal', bins=bins, ec="k")

g.axes[-1].legend()
plt.show()
```



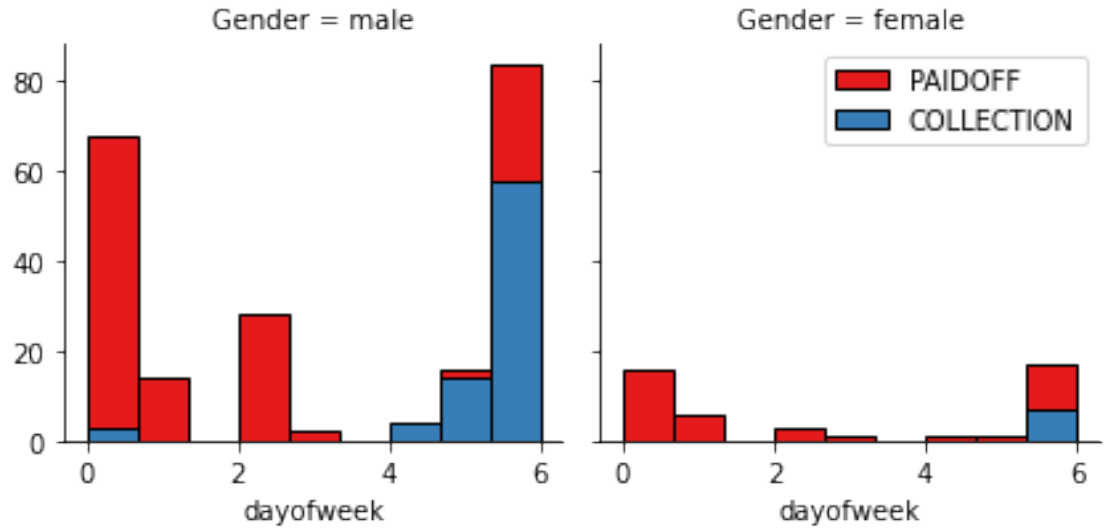
[8]:
```
bins = np.linspace(df.age.min(), df.age.max(), 10)
g = sns.FacetGrid(df, col="Gender", hue="loan_status", palette="Set1",␣
↪col_wrap=2)
g.map(plt.hist, 'age', bins=bins, ec="k")

g.axes[-1].legend()
plt.show()
```



2 Pre-processing: Feature selection/extraction

2.0.1 Let's look at the day of the week people get the loan

[9]:
```
df['dayofweek'] = df['effective_date'].dt.dayofweek
bins = np.linspace(df.dayofweek.min(), df.dayofweek.max(), 10)
g = sns.FacetGrid(df, col="Gender", hue="loan_status", palette="Set1",␣
 ↪col_wrap=2)
g.map(plt.hist, 'dayofweek', bins=bins, ec="k")
g.axes[-1].legend()
plt.show()
```



We see that people who get the loan at the end of the week don't pay it off, so let's use Feature binarization to set a threshold value less than day 4

[10]:
```
df['weekend'] = df['dayofweek'].apply(lambda x: 1 if (x>3)  else 0)
df.head()
```

[10]:

	Unnamed: 0	Unnamed: 0.1	loan_status	Principal	terms	effective_date	due_date	age	education	Gender	dayofweek	weekend
0	0	0	PAIDOFF	1000	30	2016-09-08	2016-10-07	45	High School or Below	male	3	0
1	2	2	PAIDOFF	1000	30	2016-09-08	2016-10-07	33	Bechalor	female	3	0
2	3	3	PAIDOFF	1000	15	2016-09-08	2016-09-22	27	college	male	3	0
3	4	4	PAIDOFF	1000	30	2016-09-09						
4	6	6	PAIDOFF	1000	30	2016-09-09						

```
3 2016-10-08   28              college  female          4        1
4 2016-10-08   29              college    male          4        1
```

2.1 Convert Categorical features to numerical values

Let's look at gender:

[11]:
```
df.groupby(['Gender'])['loan_status'].value_counts(normalize=True)
```

[11]:
```
Gender  loan_status
female  PAIDOFF        0.865385
        COLLECTION     0.134615
male    PAIDOFF        0.731293
        COLLECTION     0.268707
Name: loan_status, dtype: float64
```

86 % of female pay there loans while only 73 % of males pay there loan

Let's convert male to 0 and female to 1:

[12]:
```
df['Gender'].replace(to_replace=['male','female'], value=[0,1],inplace=True)
df.head()
```

[12]:
```
   Unnamed: 0  Unnamed: 0.1 loan_status  Principal  terms effective_date  \
0           0             0     PAIDOFF       1000     30     2016-09-08
1           2             2     PAIDOFF       1000     30     2016-09-08
2           3             3     PAIDOFF       1000     15     2016-09-08
3           4             4     PAIDOFF       1000     30     2016-09-09
4           6             6     PAIDOFF       1000     30     2016-09-09

    due_date  age             education  Gender  dayofweek  weekend
0 2016-10-07   45  High School or Below       0          3        0
1 2016-10-07   33              Bechalor       1          3        0
2 2016-09-22   27               college       0          3        0
3 2016-10-08   28               college       1          4        1
4 2016-10-08   29               college       0          4        1
```

2.2 One Hot Encoding

How about education?

[13]:
```
df.groupby(['education'])['loan_status'].value_counts(normalize=True)
```

[13]:
```
education             loan_status
Bechalor              PAIDOFF        0.750000
                      COLLECTION     0.250000
High School or Below  PAIDOFF        0.741722
                      COLLECTION     0.258278
Master or Above       COLLECTION     0.500000
```

```
                      PAIDOFF        0.500000
college               PAIDOFF        0.765101
                      COLLECTION     0.234899
Name: loan_status, dtype: float64
```

Features before One Hot Encoding

[14]:
```
df[['Principal','terms','age','Gender','education']].head()
```

[14]:

	Principal	terms	age	Gender	education
0	1000	30	45	0	High School or Below
1	1000	30	33	1	Bechalor
2	1000	15	27	0	college
3	1000	30	28	1	college
4	1000	30	29	0	college

Use one hot encoding technique to conver categorical varables to binary variables and append them to the feature Data Frame

[15]:
```
Feature = df[['Principal','terms','age','Gender','weekend']]
Feature = pd.concat([Feature,pd.get_dummies(df['education'])], axis=1)
Feature.drop(['Master or Above'], axis = 1,inplace=True)
Feature.head()
```

[15]:

	Principal	terms	age	Gender	weekend	Bechalor	High School or Below	college
0	1000	30	45	0	0	0	1	0
1	1000	30	33	1	0	1	0	0
2	1000	15	27	0	0	0	0	1
3	1000	30	28	1	1	0	0	1
4	1000	30	29	0	1	0	0	1

2.2.1 Feature Selection

Let's define feature sets, X:

[16]:
```
X = Feature
X[0:5]
```

[16]:

	Principal	terms	age	Gender	weekend	Bechalor	High School or Below	college
0	1000	30	45	0	0	0	1	0
1	1000	30	33	1	0	1	0	0

```
2       1000     15   27       0        0         0                    0
3       1000     30   28       1        1         0                    0
4       1000     30   29       0        1         0                    0

   college
0        0
1        0
2        1
3        1
4        1
```

What are our lables?

Need to turn load_status column from 'COLLECTION' and 'PAIDOFF' inputs to binary 1's or zeroes to get the algorithms to work

[17]:
```
df['loan_status'].value_counts()
```

[17]:
```
PAIDOFF       260
COLLECTION     86
Name: loan_status, dtype: int64
```

[18]:
```
df['loan_status'].replace(to_replace=['PAIDOFF','COLLECTION'],␣
 ↪value=[0,1],inplace=True)
df['loan_status'].value_counts()
```

[18]:
```
0    260
1     86
Name: loan_status, dtype: int64
```

[66]:
```
y = df['loan_status'].values
y.size
```

[66]:
```
346
```

2.3 Normalize Data

Data Standardization give data zero mean and unit variance (technically should be done after train test split)

[72]:
```
X= preprocessing.StandardScaler().fit(X).transform(X)
X[0:5]
X.shape
```

[72]:
```
(346, 8)
```

3 Classification

Now, it is your turn, use the training set to build an accurate model. Then use the test set to report the accuracy of the model You should use the following algorithm:

- K Nearest Neighbor(KNN)
- Decision Tree
- Support Vector Machine
- Logistic Regression

___ Notice:___

- You can go above and change the pre-processing, feature selection, feature-extraction, and so on, to make a better model.
- You should use either scikit-learn, Scipy or Numpy libraries for developing the classification algorithms.
- You should include the code of the algorithm in the following cells.

4 K Nearest Neighbor(KNN)

Notice: You should find the best k to build the model with the best accuracy.
warning: You should not use the **loan_test.csv** for finding the best k, however, you can split your train_loan.csv into train and test to find the best **k**.

[25]:
```
X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2,␣
 ↪random_state=4)
print ('Train set:', X_train.shape,  y_train.shape)
print ('Test set:', X_test.shape,  y_test.shape)
```

```
Train set: (276, 8) (276,)
Test set: (70, 8) (70,)
```

[26]:
```
X_train
```

[26]:
```
array([[ 0.51578458, -0.95911111,  0.67333883, …, -0.38170062,
        -0.87997669,  1.14984679],
       [ 0.51578458,  0.92071769, -0.81902922, …, -0.38170062,
         1.13639374, -0.86968108],
       [ 0.51578458,  0.92071769,  0.01006414, …, -0.38170062,
        -0.87997669,  1.14984679],
       …,
       [ 0.51578458, -0.95911111, -0.65321055, …, -0.38170062,
        -0.87997669,  1.14984679],
       [ 0.51578458,  0.92071769, -0.81902922, …, -0.38170062,
        -0.87997669,  1.14984679],
       [ 0.51578458,  0.92071769, -0.15575453, …, -0.38170062,
         1.13639374, -0.86968108]])
```

[27]:
```
Ks = 12
mean_acc = np.zeros((Ks-1))
std_acc = np.zeros((Ks-1))

for n in range(1,Ks):

    #Train Model and Predict
    neigh = KNeighborsClassifier(n_neighbors = n).fit(X_train,y_train)
    yhat=neigh.predict(X_test)
    mean_acc[n-1] = metrics.accuracy_score(y_test, yhat)


    std_acc[n-1]=np.std(yhat==y_test)/np.sqrt(yhat.shape[0])

mean_acc
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
```

```
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
```

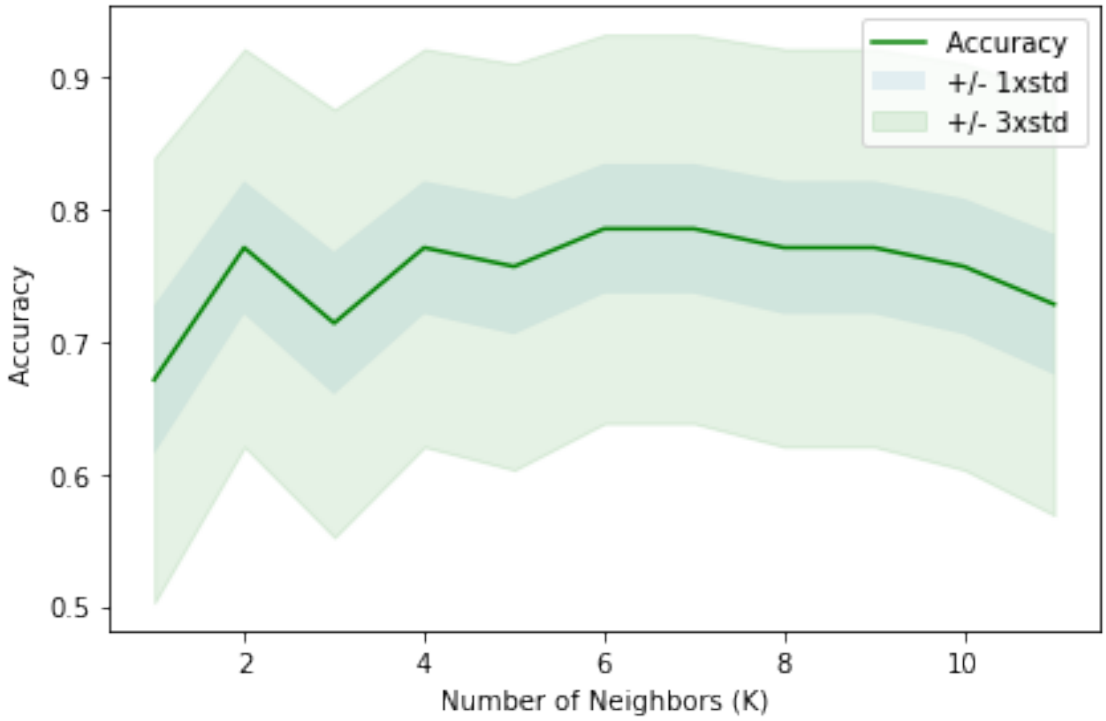
```
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
```

```
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
```

```
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
```

[27]:
```
array([0.67142857, 0.77142857, 0.71428571, 0.77142857, 0.75714286,
       0.78571429, 0.78571429, 0.77142857, 0.77142857, 0.75714286,
       0.72857143])
```

[28]:
```
plt.plot(range(1,Ks),mean_acc,'g')
plt.fill_between(range(1,Ks),mean_acc - 1 * std_acc,mean_acc + 1 * std_acc,␣
 ↪alpha=0.10)
plt.fill_between(range(1,Ks),mean_acc - 3 * std_acc,mean_acc + 3 * std_acc,␣
 ↪alpha=0.10,color="green")
plt.legend(('Accuracy ', '+/- 1xstd','+/- 3xstd'))
plt.ylabel('Accuracy ')
plt.xlabel('Number of Neighbors (K)')
plt.tight_layout()
plt.show()
```



####Testing up to k=12, the highest accuracy was achieved with both k=6 and k=7. Therefore, I will proceed with k=7 as the choice for the best k-Nearest-Neighbor model.

[29]:
```
knn = KNeighborsClassifier(n_neighbors = 7).fit(X_train,y_train)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:907: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  self._y = np.empty(y.shape, dtype=np.int)
```

[30]:
```
yhat_knn=knn.predict(X_test)
knn_accuracy = metrics.accuracy_score(y_test, yhat_knn)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
```

```
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
```

[31]:
```
knn_accuracy
```

[31]:
```
0.7857142857142857
```

5 Decision Tree

[32]:
```
loanTree = DecisionTreeClassifier(criterion="entropy", max_depth = 4)
loanTree
```

[32]:
```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=4,
            max_features=None, 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, presort=False, random_state=None,
            splitter='best')
```

[33]:
```
decision_tree = loanTree.fit(X_train,y_train)
decision_tree
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/tree/tree.py:149: DeprecationWarning: `np.int` is a deprecated
alias for the builtin `int`. To silence this warning, use `int` by itself. Doing
this will not modify any behavior and is safe. When replacing `np.int`, you may
wish to use e.g. `np.int64` or `np.int32` to specify the precision. If you wish
to review your current use, check the release note link for additional
information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  y_encoded = np.zeros(y.shape, dtype=np.int)
```

[33]:
```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=4,
            max_features=None, 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, presort=False, random_state=None,
            splitter='best')
```

[34]:
```
yhat_DecTree = decision_tree.predict(X_test)
print("DecisionTrees's Accuracy: ", metrics.accuracy_score(y_test,␣
 ↪yhat_DecTree))
```

```
DecisionTrees's Accuracy:  0.7857142857142857
```

6 Support Vector Machine

[35]:
```
clf = svm.SVC(kernel='rbf')
SVM = clf.fit(X_train, y_train)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/svm/base.py:196: FutureWarning: The default value of gamma will
change from 'auto' to 'scale' in version 0.22 to account better for unscaled
features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
  "avoid this warning.", FutureWarning)
```

[36]:
```
SVM
```

[36]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
  kernel='rbf', max_iter=-1, probability=False, random_state=None,
  shrinking=True, tol=0.001, verbose=False)
```

7 Logistic Regression

[37]:
```
lr = LogisticRegression(C=0.01, solver='liblinear').fit(X_train,y_train)
lr
```

[37]:
```
LogisticRegression(C=0.01, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='liblinear',
          tol=0.0001, verbose=0, warm_start=False)
```

[38]:
```
yhat_lr = lr.predict(X_test)
yhat_lr
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/base.py:283: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  indices = (scores > 0).astype(np.int)
```

[38]:
```
array([1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0])
```

[39]:
```
yhat_prob_log = lr.predict_proba(X_test)
yhat_prob_log
```

[39]:
```
array([[0.4965762 , 0.5034238 ],
       [0.54793889, 0.45206111],
       [0.69185868, 0.30814132],
       [0.65740572, 0.34259428],
       [0.67974106, 0.32025894],
       [0.68319463, 0.31680537],
       [0.51169815, 0.48830185],
       [0.52176927, 0.47823073],
       [0.65740572, 0.34259428],
       [0.5065944 , 0.4934056 ],
       [0.66193294, 0.33806706],
       [0.50337769, 0.49662231],
       [0.75108093, 0.24891907],
       [0.6580905 , 0.3419095 ],
       [0.56248211, 0.43751789],
       [0.74239503, 0.25760497],
       [0.47642812, 0.52357188],
       [0.69549722, 0.30450278],
       [0.49833637, 0.50166363],
       [0.6804029 , 0.3195971 ],
       [0.55723012, 0.44276988],
       [0.50589815, 0.49410185],
       [0.48649667, 0.51350333],
       [0.52796502, 0.47203498],
       [0.59055306, 0.40944694],
       [0.49153558, 0.50846442],
       [0.48901585, 0.51098415],
       [0.62542353, 0.37457647],
       [0.49581577, 0.50418423],
       [0.74700365, 0.25299635],
       [0.53175887, 0.46824113],
       [0.53975312, 0.46024688],
       [0.53793083, 0.46206917],
       [0.51597575, 0.48402425],
       [0.61181809, 0.38818191],
       [0.54178674, 0.45821326],
       [0.49833637, 0.50166363],
       [0.71026415, 0.28973585],
       [0.5430118 , 0.4569882 ],
       [0.54505282, 0.45494718],
       [0.49329538, 0.50670462],
       [0.67820638, 0.32179362],
       [0.54754224, 0.45245776],
       [0.49153558, 0.50846442],
```

```
       [0.69335769, 0.30664231],
       [0.50484416, 0.49515584],
       [0.52924756, 0.47075244],
       [0.50337769, 0.49662231],
       [0.54428875, 0.45571125],
       [0.54432377, 0.45567623],
       [0.72205941, 0.27794059],
       [0.53255135, 0.46744865],
       [0.69498919, 0.30501081],
       [0.51093806, 0.48906194],
       [0.71941574, 0.28058426],
       [0.75078894, 0.24921106],
       [0.68477194, 0.31522806],
       [0.56963005, 0.43036995],
       [0.53175887, 0.46824113],
       [0.66486368, 0.33513632],
       [0.58074774, 0.41925226],
       [0.66866833, 0.33133167],
       [0.54178674, 0.45821326],
       [0.47391365, 0.52608635],
       [0.67600195, 0.32399805],
       [0.50589815, 0.49410185],
       [0.66866833, 0.33133167],
       [0.58262074, 0.41737926],
       [0.55003892, 0.44996108],
       [0.67600195, 0.32399805]])
```

8 Model Evaluation using Test set

First, download and load the test set:

[40]:
```
!wget -O loan_test.csv https://s3-api.us-geo.objectstorage.softlayer.net/
↪cf-courses-data/CognitiveClass/ML0101ENv3/labs/loan_test.csv
```

```
--2022-05-31 18:01:42--  https://s3-api.us-geo.objectstorage.softlayer.net/cf-
courses-data/CognitiveClass/ML0101ENv3/labs/loan_test.csv
Resolving s3-api.us-geo.objectstorage.softlayer.net (s3-api.us-
geo.objectstorage.softlayer.net)… 67.228.254.196
Connecting to s3-api.us-geo.objectstorage.softlayer.net (s3-api.us-
geo.objectstorage.softlayer.net)|67.228.254.196|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 3642 (3.6K) [text/csv]
Saving to: ‘loan_test.csv’

loan_test.csv       100%[===================>]   3.56K  5.57KB/s    in 0.6s

2022-05-31 18:01:42 (5.57 KB/s) - ‘loan_test.csv’ saved [3642/3642]
```

8.0.1 Load Test set for evaluation

[48]:
```
testing_data_df = pd.read_csv('loan_test.csv')
testing_data_df.head()
```

[48]:

	Unnamed: 0	Unnamed: 0.1	loan_status	Principal	terms	effective_date	due_date	age	education	Gender
0	1	1	PAIDOFF	1000	30	9/8/2016	10/7/2016	50	Bechalor	female
1	5	5	PAIDOFF	300	7	9/9/2016	9/15/2016	35	Master or Above	male
2	21	21	PAIDOFF	1000	30	9/10/2016	10/9/2016	43	High School or Below	female
3	24	24	PAIDOFF	1000	30	9/10/2016	10/9/2016	26	college	male
4	35	35	PAIDOFF	800	15	9/11/2016	9/25/2016	29	Bechalor	male

[49]:
```
testing_data_df['due_date'] = pd.to_datetime(testing_data_df['due_date'])
testing_data_df['effective_date'] = pd.
 ↪to_datetime(testing_data_df['effective_date'])
testing_data_df['dayofweek'] = testing_data_df['effective_date'].dt.dayofweek
testing_data_df['Gender'].replace(to_replace=['male','female'],␣
 ↪value=[0,1],inplace=True)
testing_data_df['weekend'] = testing_data_df['dayofweek'].apply(lambda x: 1 if␣
 ↪(x>3)  else 0)
testing_data_df.head()
```

[49]:

	Unnamed: 0	Unnamed: 0.1	loan_status	Principal	terms	effective_date	due_date	age	education	Gender	dayofweek	weekend
0	1	1	PAIDOFF	1000	30	2016-09-08	2016-10-07	50	Bechalor	1	3	0
1	5	5	PAIDOFF	300	7	2016-09-09	2016-09-15	35	Master or Above	0	4	1
2	21	21	PAIDOFF	1000	30	2016-09-10	2016-10-09	43	High School or Below	1	5	1
3	24	24	PAIDOFF	1000	30	2016-09-10	2016-10-09	26	college	0	5	1
4	35	35	PAIDOFF	800	15	2016-09-11	2016-09-25	29	Bechalor	0	6	1

[50]:
```
testing_data_df['loan_status'].replace(to_replace=['PAIDOFF','COLLECTION'],␣
 ↪value=[0,1],inplace=True)
```

```
testing_data_df['loan_status'].value_counts()
```

[50]:
```
0    40
1    14
Name: loan_status, dtype: int64
```

[51]:
```
Feature_testing =␣
 ↪testing_data_df[['Principal','terms','age','Gender','weekend']]
Feature_testing = pd.concat([Feature_testing,pd.
 ↪get_dummies(testing_data_df['education'])], axis=1)
Feature_testing.drop(['Master or Above'], axis = 1,inplace=True)
Feature_testing.head()
```

[51]:

	Principal	terms	age	Gender	weekend	Bechalor	High School or Below	college
0	1000	30	50	1	0	1	0	0
1	300	7	35	0	1	0	0	0
2	1000	30	43	1	1	0	1	0
3	1000	30	26	0	1	0	0	1
4	800	15	29	0	1	1	0	0

[76]:
```
X_testing = Feature_testing
X_testing[0:5]
```

[76]:

	Principal	terms	age	Gender	weekend	Bechalor	High School or Below	college
0	1000	30	50	1	0	1	0	0
1	300	7	35	0	1	0	0	0
2	1000	30	43	1	1	0	1	0
3	1000	30	26	0	1	0	0	1
4	800	15	29	0	1	1	0	0

[77]:
```
X_testing= preprocessing.StandardScaler().fit(X_testing).transform(X_testing)
X_testing[0:5]
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/preprocessing/data.py:625: DataConversionWarning: Data with
input dtype uint8, int64 were all converted to float64 by StandardScaler.
  return self.partial_fit(X, y)
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/ipykernel_launcher.py:1: DataConversionWarning: Data with input dtype
uint8, int64 were all converted to float64 by StandardScaler.
  """Entry point for launching an IPython kernel.
```

[77]:
```
array([[ 0.49362588,  0.92844966,  3.05981865,  1.97714211, -1.30384048,
         2.39791576, -0.79772404, -0.86135677],
       [-3.56269116, -1.70427745,  0.53336288, -0.50578054,  0.76696499,
        -0.41702883, -0.79772404, -0.86135677],
       [ 0.49362588,  0.92844966,  1.88080596,  1.97714211,  0.76696499,
        -0.41702883,  1.25356634, -0.86135677],
       [ 0.49362588,  0.92844966, -0.98251057, -0.50578054,  0.76696499,
        -0.41702883, -0.79772404,  1.16095912],
       [-0.66532184, -0.78854628, -0.47721942, -0.50578054,  0.76696499,
         2.39791576, -0.79772404, -0.86135677]])
```

[78]:
```
y_testing = testing_data_df['loan_status'].values
y_testing[0:5]
print ('Test set:', X_testing.shape,  y_testing.shape)
```

```
Test set: (54, 8) (54,)
```

9 Report

You should be able to report the accuracy of the built model using different evaluation metrics:

Algorithm	Jaccard	F1-score	LogLoss
KNN	?	?	NA
Decision Tree	?	?	NA
SVM	?	?	NA
LogisticRegression	?	?	?

[79]:
```
knn_applied = knn.predict(X_testing)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/neighbors/base.py:442: DeprecationWarning: distutils Version
classes are deprecated. Use packaging.version instead.
  old_joblib = LooseVersion(joblib_version) < LooseVersion('0.12')
```

[80]:
```
knn_jaccard = jaccard_similarity_score(y_testing, knn_applied)
knn_jaccard
```

[80]:
```
0.6666666666666666
```

[81]:
```
knn_f1 = f1_score(y_testing, knn_applied, average='weighted')
knn_f1
```

[81]:
```
0.6328400281888654
```

[82]:
```
dec_tree_applied = decision_tree.predict(X_testing)
```

[83]:
```
dec_tree_jaccard = jaccard_similarity_score(y_testing, dec_tree_applied)
dec_tree_jaccard
```

[83]:
```
0.7592592592592593
```

[84]:
```
dec_tree_f1 = f1_score(y_testing, dec_tree_applied, average='weighted')
dec_tree_f1
```

[84]:
```
0.6717642373556352
```

[85]:
```
SVM_applied = SVM.predict(X_testing)
```

[86]:
```
SVM_jaccard = jaccard_similarity_score(y_testing, SVM_applied)
SVM_jaccard
```

[86]:
```
0.7962962962962963
```

[87]:
```
SVM_f1 = f1_score(y_testing, SVM_applied, average='weighted')
SVM_f1
```

[87]:
```
0.7583503077293734
```

[88]:
```
lr_applied = lr.predict(X_testing)
```

```
/home/jupyterlab/conda/envs/python/lib/python3.7/site-
packages/sklearn/linear_model/base.py:283: DeprecationWarning: `np.int` is a
deprecated alias for the builtin `int`. To silence this warning, use `int` by
itself. Doing this will not modify any behavior and is safe. When replacing
`np.int`, you may wish to use e.g. `np.int64` or `np.int32` to specify the
precision. If you wish to review your current use, check the release note link
for additional information.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
  indices = (scores > 0).astype(np.int)
```

[89]:
```
lr_jaccard = jaccard_similarity_score(y_testing, lr_applied)
lr_jaccard
```

[89]:
```
0.7407407407407407
```

[90]:
```
lr_f1 = f1_score(y_testing, lr_applied, average='weighted')
lr_f1
```

[90]:
```
0.6604267310789049
```

[93]:
```
lr_log_loss = log_loss(y_testing, lr.predict_proba(X_testing))
lr_log_loss
```

[93]:
```
0.5672153379912981
```

[95]:
```
data = [ ['KNN', knn_jaccard, knn_f1, 'N/A'], ['Decision Tree',␣
 ↪dec_tree_jaccard, dec_tree_f1, 'N/A'], ['SVM', SVM_jaccard, SVM_f1, 'N/A'],␣
 ↪['Logistic Regression', lr_jaccard, lr_f1, lr_log_loss] ]
results = pd.DataFrame(data, columns=['Algorithm', 'Jaccard', 'F1-Score',␣
 ↪'LogLoss'])
results
```

[95]:

	Algorithm	Jaccard	F1-Score	LogLoss
0	KNN	0.666667	0.632840	N/A
1	Decision Tree	0.759259	0.671764	N/A
2	SVM	0.796296	0.758350	N/A
3	Logistic Regression	0.740741	0.660427	0.567215

Want to learn more?

IBM SPSS Modeler is a comprehensive analytics platform that has many machine learning algorithms. It has been designed to bring predictive intelligence to decisions made by individuals, by groups, by systems – by your enterprise as a whole. A free trial is available through this course, available here: SPSS Modeler

Also, you can use Watson Studio to run these notebooks faster with bigger datasets. Watson Studio is IBM's leading cloud solution for data scientists, built by data scientists. With Jupyter notebooks, RStudio, Apache Spark and popular libraries pre-packaged in the cloud, Watson Studio enables data scientists to collaborate on their projects without having to install anything. Join the fast-growing community of Watson Studio users today with a free account at Watson Studio

Thanks for completing this lesson!

Author: Saeed Aghabozorgi

Saeed Aghabozorgi, PhD is a Data Scientist in IBM with a track record of developing enterprise level applications that substantially increases clients' ability to turn data into actionable knowledge. He is a researcher in data mining field and expert in developing advanced analytic methods like machine learning and statistical modelling on large datasets.

9.1 Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-10-27	2.1	Lakshmi Holla	Made changes in import statement due to updates in version of sklearn library
2020-08-27	2.0	Malika Singla	Added lab to GitLab

##