CSE_489_ACP_DL

October 3, 2020

[56]:

```
# Avoiding warning
import warnings
def warn(*args, **kwargs): pass
warnings.warn = warn
# ______
```

[57]:

```
import numpy as np
```

[58]:

```
inputFile = '/home/hbins413/Desktop/acp740.txt'
```

[59]:

```
def readFASTAs(fileName):
    '''
    :param fileName:
    :return: genome sequences
    '''
    with open(fileName, 'r') as file:
        v = []
        genome = ''
        for line in file:
            if line[0] != '>':
                genome += line.strip()
            else:
                v.append(genome.upper())
                genome = ''
        v.append(genome.upper())
        del v[0]
        return v
```

[60]:

```
Sequences = readFASTAs(inputFile)

X = []
for seq in Sequences:
    X.append([seq.count('A'), seq.count('C'), seq.count('D'), seq.count('E'),␣
 ↪seq.count('F'), seq.count('G'), seq.count('H'), seq.count('I'), seq.
 ↪count('K'), seq.count('L'), seq.count('M'), seq.count('N'), seq.count('P'),␣
 ↪seq.count('Q'), seq.count('R'), seq.count('S'), seq.count('T'), seq.
 ↪count('V'), seq.count('W'), seq.count('Y')])
```

```
X = np.array(X)
```

[61]:

```
X
```

[61]:

```
array([[11,  0,  0, ...,  3,  1,  0],
       [ 1,  0,  1, ...,  0,  0,  0],
       [ 1,  0,  1, ...,  3,  0,  0],
       ...,
       [ 2,  2,  0, ...,  2,  0,  3],
       [ 4,  4,  3, ...,  2,  0,  0],
       [ 1,  0,  1, ...,  0,  0,  0]])
```

[62]:

```
print(X.shape)
```

```
(740, 20)
```

[63]:

```
Y  = [1 for _ in range(376)]
Y += [0 for _ in range(364)]
```

[64]:

```
# Core:
import pandas as pd

# Machine Learning Algorithms"

from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier


# Dataset Handle
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

# Performance:
from sklearn.metrics import (confusion_matrix, accuracy_score,␣
 ↪classification_report)
```

[65]:

```
# Xtrain, Xtest, Ytrain, Ytest = train_test_split(X, Y, test_size=0.25)
```

[66]:

```
# # Scalling
# scaling = StandardScaler()
# Xtrain = scaling.fit_transform(Xtrain)
# Xtest  = scaling.transform(Xtest)
```

[67]:

```
classifiers = [
    LogisticRegression(),
    KNeighborsClassifier(n_neighbors=5),
    DecisionTreeClassifier(),
    SVC(kernel='rbf', probability=True),
    GaussianNB(),
    RandomForestClassifier()
]
```

[70]:

```
for classifier in classifiers:
    model = classifier
#     model.fit(Xtrain, Ytrain)




#     Task-1: 70/30
#     Yp = model.predict(Xtest)
#     accuracy = accuracy_score(y_true=Ytest, y_pred=Yp)
#     print('Classifier: {}, Accuracy: {:0.2f}'.format(classifier.__class__.
 ↪__name__, accuracy))

#     Task-2: cv=5

    scaling = StandardScaler()
    X = scaling.fit_transform(X)
    accuracy = cross_val_score(model, X, Y, cv=5)
    accuracy = np.mean(accuracy)
    print('Classifier: {}, Accuracy: {:0.2f}'.format(classifier.__class__.
 ↪__name__, accuracy))
```

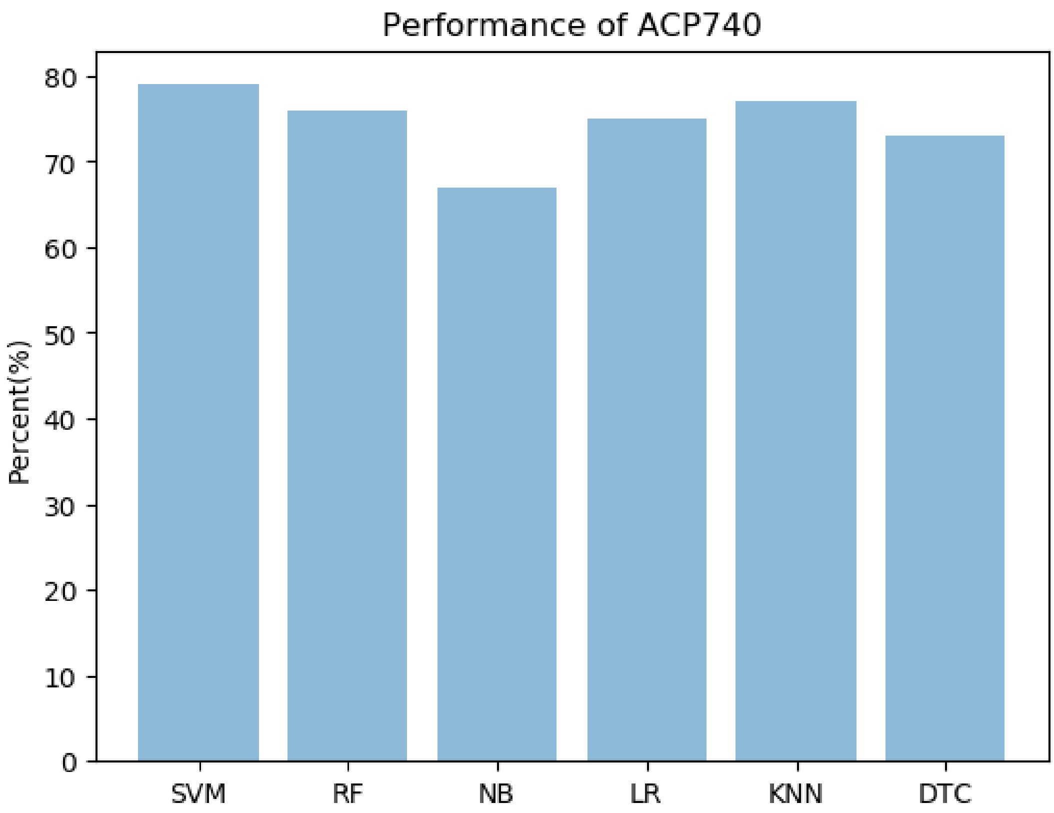
```
Classifier: LogisticRegression, Accuracy: 0.75
Classifier: KNeighborsClassifier, Accuracy: 0.77
Classifier: DecisionTreeClassifier, Accuracy: 0.73
Classifier: SVC, Accuracy: 0.79
Classifier: GaussianNB, Accuracy: 0.67
Classifier: RandomForestClassifier, Accuracy: 0.76
```

```
/home/hbins413/anaconda3/lib/python3.7/site-
packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value of
n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/home/hbins413/anaconda3/lib/python3.7/site-
packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value of
n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/home/hbins413/anaconda3/lib/python3.7/site-
```

```
packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value of
n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/home/hbins413/anaconda3/lib/python3.7/site-
packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value of
n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
/home/hbins413/anaconda3/lib/python3.7/site-
packages/sklearn/ensemble/forest.py:245: FutureWarning: The default value of
n_estimators will change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

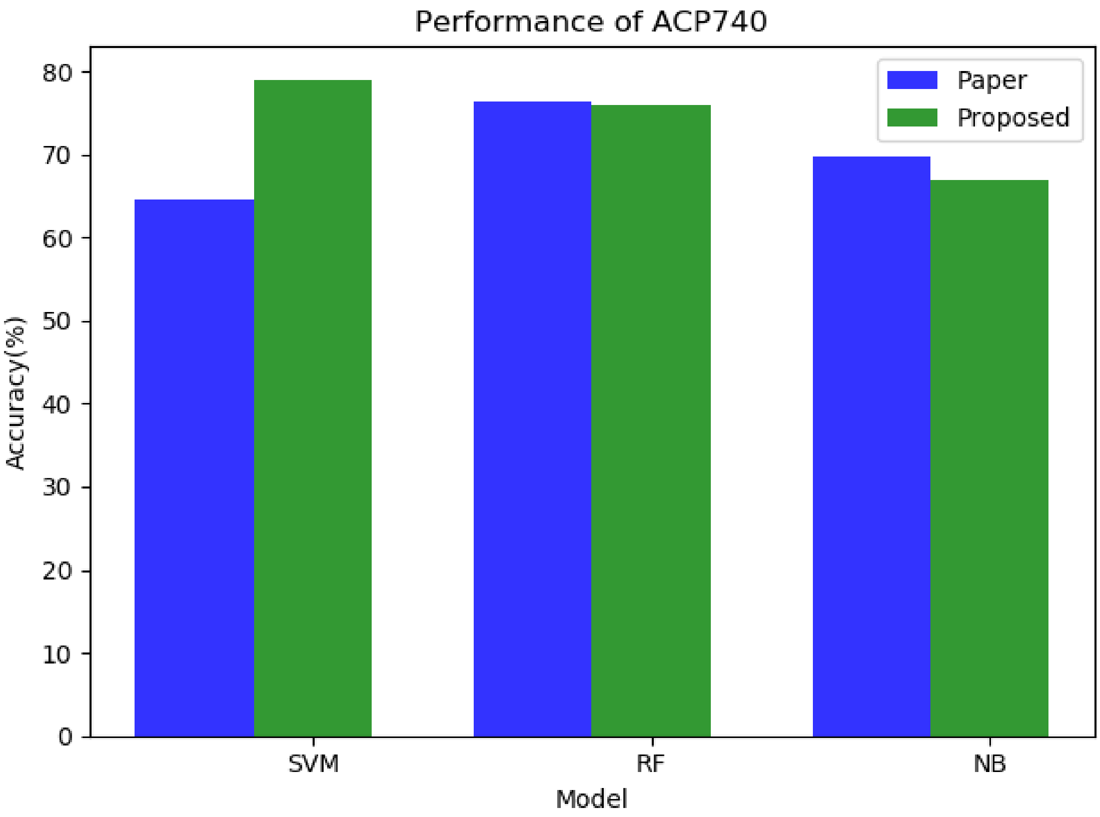
[74]:

```python
import matplotlib.pyplot as plt; plt.rcdefaults()
import numpy as np
import matplotlib.pyplot as plt

objects = ('SVM', 'RF', 'NB', 'LR', 'KNN', 'DTC')
y_pos = np.arange(len(objects))
performance = [79,76,67,75,77,73]

plt.bar(y_pos, performance, align='center', alpha=0.5)
plt.xticks(y_pos, objects)
plt.ylabel('Percent(%)')
plt.title('Performance of ACP740')

plt.show()
```



[79]:

```
import matplotlib.pyplot as plt

# data to plot
n_groups = 3
means_frank = (64.59, 76.36, 69.73)
means_guido = (79, 76, 67)

# create plot
fig, ax = plt.subplots()
index = np.arange(n_groups)
bar_width = 0.35
opacity = 0.8

rects1 = plt.bar(index, means_frank, bar_width,
alpha=opacity,
color='b',
label='Paper')

rects2 = plt.bar(index + bar_width, means_guido, bar_width,
alpha=opacity,
color='g',
label='Proposed')

plt.xlabel('Model')
plt.ylabel('Accuracy(%)')
plt.title('Performance of ACP740')
plt.xticks(index + bar_width, ('SVM', 'RF', 'NB', 'D'))
plt.legend()

plt.tight_layout()
plt.show()
```



```
[ ]:
```