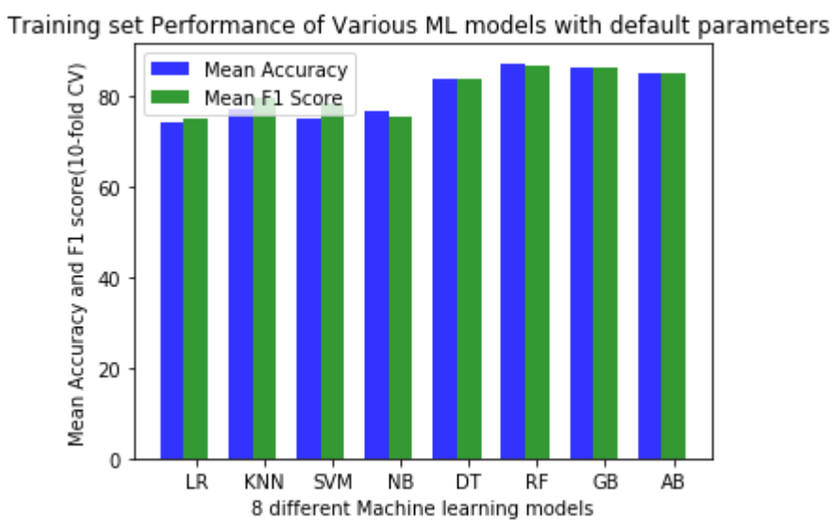
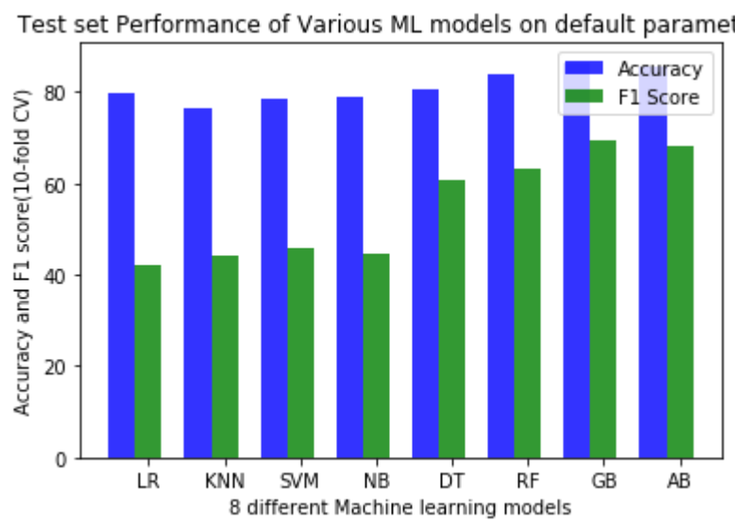
Code for graph in Section 4.1 a) - (Training set performance of 8 different machine learning algorithms)

In [6]:
```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt

# data to plot
n_groups = 8
means_accuracy = (74.37, 77.38, 75.13, 76.61, 84.00, 87.31, 86.43, 85.16)
means_f1 = (75.31, 79.90, 78.61, 75.56, 83.77, 86.72, 86.51, 85.29)

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

rects1 = plt.bar(index, means_accuracy, bar_width,alpha=opacity,color='b',label='Mean Accuracy')
rects2 = plt.bar(index + bar_width, means_f1, bar_width,alpha=opacity,color='g',label='Mean F1 Scor
e')

plt.xlabel('8 different Machine learning models')
plt.ylabel('Mean Accuracy and F1 score(10-fold CV)')
plt.title('Training set Performance of Various ML models with default parameters ')
plt.xticks(index + bar_width, ('LR', 'KNN', 'SVM', 'NB', 'DT', 'RF','GB', 'AB'))
plt.legend()

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



Code for the graph in section 4.1 b) (Test set performance of 8 different machine learning algorithms)

In [7]:
```
import numpy as np
import matplotlib.pyplot as plt

# data to plot
n_groups = 8
means_accuracy = (79.69, 76.47, 78.43, 79.10, 80.42, 83.77, 86.41, 85.63)
means_f1 = (41.95, 44.02, 45.67, 44.54, 60.72, 63.26, 69.20, 68.05)

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

rects1 = plt.bar(index, means_accuracy, bar_width,alpha=opacity,color='b',label='Accuracy')
rects2 = plt.bar(index + bar_width, means_f1, bar_width,alpha=opacity,color='g',label='F1 Score')

plt.xlabel('8 different Machine learning models')
plt.ylabel('Accuracy and F1 score(10-fold CV)')
plt.title('Test set Performance of Various ML models on default parameters ')
plt.xticks(index + bar_width, ('LR', 'KNN', 'SVM', 'NB', 'DT', 'RF','GB', 'AB'))
plt.legend()

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



Code for graph in section 4.2 a) - Training set performance of top models before and after parameter tuning

In [19]:
```
import numpy as np
import matplotlib.pyplot as plt

# data to plot
n_groups = 2
before_tuning_f1 = (86.72, 86.51)
after_tuning_f1 = (89.65, 86.94)

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

rects1 = plt.bar(index, before_tuning_f1, bar_width,alpha=opacity,color='b',label='Default Hyper-par
ameters')

rects2 = plt.bar(index + bar_width, after_tuning_f1, bar_width,alpha=opacity,color='darkorange',labe
l='Optimal Hyper-parameters')

plt.xlabel('Top 2 ML Models')
plt.ylabel('Best Mean F1 score(10-fold CV)')
plt.title('Training set Performance of top ML models before and after Hyper-parameter Optimization')
plt.xticks(index + bar_width, ('Random Forest', 'Gradient Boosting'))
plt.legend()

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