```
!pip install mlxtend --upgrade
```

```
Requirement already satisfied: mlxtend in /usr/local/lib/python3.7/dist-packages (0.14.0
Collecting mlxtend
  Downloading mlxtend-0.19.0-py2.py3-none-any.whl (1.3 MB)
     |████████████████████████████████| 1.3 MB 6.8 MB/s
Requirement already satisfied: numpy>=1.16.2 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: joblib>=0.13.2 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: scipy>=1.2.1 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: matplotlib>=3.0.0 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: pandas>=0.24.2 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: scikit-learn>=0.20.3 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: python-dateutil>=2.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in /usr/local/li
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-pac
Installing collected packages: mlxtend
  Attempting uninstall: mlxtend
    Found existing installation: mlxtend 0.14.0
    Uninstalling mlxtend-0.14.0:
      Successfully uninstalled mlxtend-0.14.0
Successfully installed mlxtend-0.19.0
```

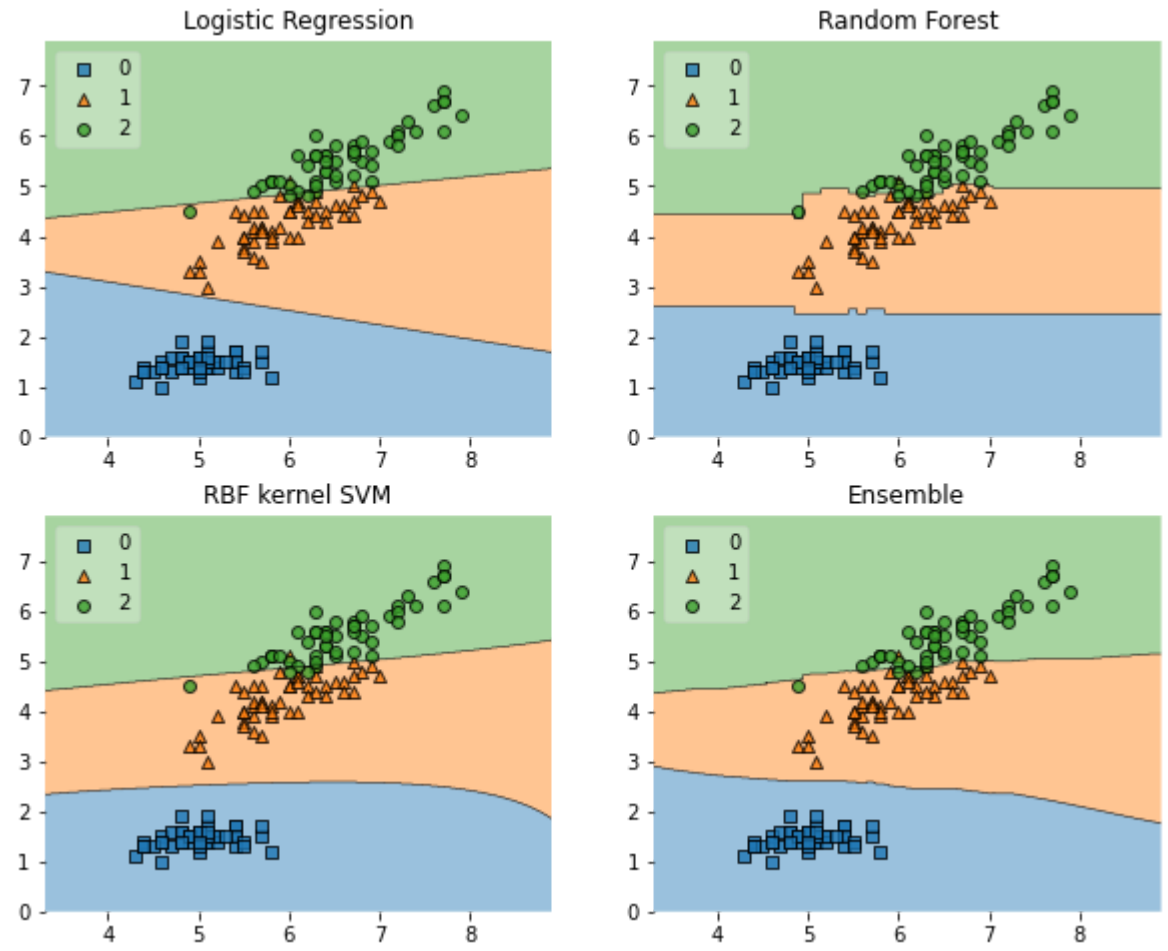
```
import mlxtend
from mlxtend.evaluate import bias_variance_decomp
from sklearn.tree import DecisionTreeClassifier
from mlxtend.data import iris_data
from sklearn.model_selection import train_test_split

# Get Data Set
X,·y·=·iris_data()
X_train_ds,·X_test_ds,·y_train_ds,·y_test_ds·=·train_test_split(X,·y,
····················································test_size=0.3,
····················································random_state=123,
····················································shuffle=True,
····················································stratify=y)

# Define Algorithm
tree = DecisionTreeClassifier(random_state=123)

# Get Bias and Variance - bias_variance_decomp function
avg_expected_loss, avg_bias, avg_var = bias_variance_decomp(
        tree, X_train_ds, y_train_ds, X_test_ds, y_test_ds,
        loss='0-1_loss',
        random_seed=123,
        num_rounds=1000)
```

```
24
25   # Display Bias and Variance
26   print(f'Average Expected Loss: {round(avg_expected_loss, 4)}n')
27   print(f'Average Bias: {round(avg_bias, 4)}')
28   print(f'Average Variance: {round(avg_var, 4)}')
```

```
Average Expected Loss: 0.0607n
Average Bias: 0.0222
Average Variance: 0.0393
```

```
 1 import numpy as np
 2 import matplotlib.pyplot as plt
 3 import matplotlib.gridspec as gridspec
 4 import itertools
 5 from sklearn.linear_model import LogisticRegression
 6 from sklearn.svm import SVC
 7 from sklearn.ensemble import RandomForestClassifier
 8 from mlxtend.classifier import EnsembleVoteClassifier
 9 from mlxtend.data import iris_data
10 from mlxtend.plotting import plot_decision_regions
11
12 # Initializing Classifiers
13 clf1 = LogisticRegression(solver='lbfgs', max_iter=1000, random_state=0)
14 clf2 = RandomForestClassifier(random_state=0)
15 clf3 = SVC(random_state=0, probability=True)
16 eclf = EnsembleVoteClassifier(clfs=[clf1, clf2, clf3],
17                               weights=[2, 1, 1], voting='soft')
18
19 # Loading some example data
20 X, y = iris_data()
21 X = X[:,[0, 2]]
22
23 # Plotting Decision Regions
24
25 gs = gridspec.GridSpec(2, 2)
26 fig = plt.figure(figsize=(10, 8))
27
28 labels = ['Logistic Regression',
29           'Random Forest',
30           'RBF kernel SVM',
31           'Ensemble']
32
33 for clf, lab, grd in zip([clf1, clf2, clf3, eclf],
34                          labels,
35                          itertools.product([0, 1],
36                          repeat=2)):
37     clf.fit(X, y)
38     ax = plt.subplot(gs[grd[0], grd[1]])
39     fig = plot_decision_regions(X=X, y=y,
40                                 clf=clf, legend=2)
```

```
41     plt.title(lab)
```




## Logistic Regression

```
 1 # Get Bias and Variance - bias_variance_decomp function
 2 avg_expected_loss, avg_bias, avg_var = bias_variance_decomp(
 3         clf1, X_train_ds, y_train_ds, X_test_ds, y_test_ds,
 4          loss='0-1_loss',
 5          random_seed=123,
 6          num_rounds=1000
 7          )
 8
 9 # Display Bias and Variance
10 print(f'Average Expected Loss: {round(avg_expected_loss, 4)}n')
11 print(f'Average Bias: {round(avg_bias, 4)}')
12 print(f'Average Variance: {round(avg_var, 4)}')
```

```
Average Expected Loss: 0.0343n
Average Bias: 0.0222
Average Variance: 0.0172
```

## Random Forest

```
# Get Bias and Variance - bias_variance_decomp function
avg_expected_loss, avg_bias, avg_var = bias_variance_decomp(
       clf2, X_train_ds, y_train_ds, X_test_ds, y_test_ds,
        loss='0-1_loss',
        random_seed=123,
        num_rounds=1000
        )

# Display Bias and Variance
print(f'Average Expected Loss: {round(avg_expected_loss, 4)}n')
print(f'Average Bias: {round(avg_bias, 4)}')
print(f'Average Variance: {round(avg_var, 4)}')
```

```
Average Expected Loss: 0.0454n
Average Bias: 0.0222
Average Variance: 0.0233
```

## SVM/RBF

```
# Get Bias and Variance - bias_variance_decomp function
avg_expected_loss, avg_bias, avg_var = bias_variance_decomp(
       clf3, X_train_ds, y_train_ds, X_test_ds, y_test_ds,
        loss='0-1_loss',
        random_seed=123,
        num_rounds=1000
        )

# Display Bias and Variance
print(f'Average Expected Loss: {round(avg_expected_loss, 4)}n')
print(f'Average Bias: {round(avg_bias, 4)}')
print(f'Average Variance: {round(avg_var, 4)}')
```

```
Average Expected Loss: 0.045n
Average Bias: 0.0444
Average Variance: 0.0284
```

## Ensemble Vote Classifier

```
  # Get Bias and Variance - bias_variance_decomp function
  avg_expected_loss, avg_bias, avg_var = bias_variance_decomp(
         eclf, X_train_ds, y_train_ds, X_test_ds, y_test_ds,
          loss='0-1_loss',
          random_seed=123,
          num_rounds=1000
          )

  # Display Bias and Variance
  print(f'Average Expected Loss: {round(avg_expected_loss, 4)}n')
```

```
print(f'Average Bias: {round(avg_bias, 4)}')
print(f'Average Variance: {round(avg_var, 4)}')
```

```
Average Expected Loss: 0.0409n
Average Bias: 0.0222
Average Variance: 0.0189
```

## How to Select a Model with LazyPredict

```
!pip install lazypredict
```

```
Requirement already satisfied: lazypredict in /usr/local/lib/python3.7/dist-packages (0
Requirement already satisfied: click==7.1.2 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: lightgbm==2.3.1 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: pytest==5.4.3 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: numpy==1.19.1 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: xgboost==1.1.1 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: scipy==1.5.4 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: PyYAML==5.3.1 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: six==1.15.0 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: scikit-learn==0.23.1 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: joblib==1.0.0 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: pandas==1.0.5 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: tqdm==4.56.0 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: python-dateutil>=2.6.1 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: attrs>=17.4.0 in /usr/local/lib/python3.7/dist-packages (
Requirement already satisfied: more-itertools>=4.0.0 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: pluggy<1.0,>=0.12 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: importlib-metadata>=0.12 in /usr/local/lib/python3.7/dist
Requirement already satisfied: py>=1.5.0 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: wcwidth in /usr/local/lib/python3.7/dist-packages (from p
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/dist
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist
```

```
### importing lazypredict library
import lazypredict
### importing LazyClassifier for classification problem
from lazypredict.Supervised import LazyClassifier
### spliting dataset into training and testing part
from sklearn.model_selection import train_test_split
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:143: FutureWarning:
  warnings.warn(message, FutureWarning)
```

```
from mlxtend.data import iris_data
```

```
X, y = iris_data()
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3,
                                                    random_state=123,
                                                    shuffle=True,
                                                    stratify=y)
### splitting dataset into training and testing part(50% training and 50% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=.5, random_state =123
```

```
clf = LazyClassifier(verbose=0, ignore_warnings=True, custom_metric = None)
```

```
  ### fitting data in LazyClassifier
  models,predictions = clf.fit(X_train, X_test, y_train, y_test)
  ### lets check which model did better on Breast Cancer Dataset
  from pandas import DataFrame
  import pandas as pd
  data=pd.DataFrame(models)
  data
```

```
100%|██████████| 29/29 [00:02<00:00, 10.51it/s]
```

| Model | Accuracy | Balanced Accuracy | ROC AUC | F1 Score | Time Taken |
|---|---|---|---|---|---|
| LinearDiscriminantAnalysis | 0.99 | 0.99 | None | 0.99 | 0.06 |
| AdaBoostClassifier | 0.97 | 0.98 | None | 0.97 | 0.46 |
| LogisticRegression | 0.97 | 0.98 | None | 0.97 | 0.03 |
| GaussianNB | 0.97 | 0.98 | None | 0.97 | 0.03 |
| PassiveAggressiveClassifier | 0.97 | 0.98 | None | 0.97 | 0.03 |
| RandomForestClassifier | 0.96 | 0.96 | None | 0.96 | 0.33 |
| ExtraTreeClassifier | 0.96 | 0.96 | None | 0.96 | 0.03 |
| QuadraticDiscriminantAnalysis | 0.96 | 0.96 | None | 0.96 | 0.02 |
| SGDClassifier | 0.96 | 0.96 | None | 0.96 | 0.02 |
| Perceptron | 0.96 | 0.96 | None | 0.96 | 0.04 |
| XGBClassifier | 0.95 | 0.95 | None | 0.95 | 0.63 |
| DecisionTreeClassifier | 0.95 | 0.95 | None | 0.95 | 0.03 |
| ExtraTreesClassifier | 0.95 | 0.95 | None | 0.95 | 0.26 |
| BaggingClassifier | 0.95 | 0.95 | None | 0.95 | 0.05 |
| LinearSVC | 0.95 | 0.95 | None | 0.95 | 0.05 |
| CalibratedClassifierCV | 0.95 | 0.95 | None | 0.95 | 0.10 |
| LGBMClassifier | 0.95 | 0.95 | None | 0.95 | 0.20 |
| NuSVC | 0.93 | 0.94 | None | 0.93 | 0.03 |
| KNeighborsClassifier | 0.93 | 0.94 | None | 0.93 | 0.03 |
| SVC | 0.93 | 0.94 | None | 0.93 | 0.02 |
| RidgeClassifierCV | 0.91 | 0.91 | None | 0.91 | 0.02 |

✓ 3s completed at 4:36 PM