```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score


df = pd.read_csv("/content/diabetes.csv")


df.head()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigre
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	

Next steps: View recommended plots

```
X = df.iloc[:, [1,6]].values
y = df.iloc[:, 8].values
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)

clf = RandomForestClassifier(n_estimators=75)
clf.fit(X_train, y_train)

y_pred = clf.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.7447916666666666
```

```
from sklearn.ensemble import VotingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier


df
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedig
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

Next steps: View recommended plots

```
X = df.iloc[:, [1,6]].values
y = df.iloc[:, 8].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25)


log_reg = LogisticRegression(random_state=42)
dt = DecisionTreeClassifier(random_state=42)
knn = KNeighborsClassifier()


from sklearn.naive_bayes import GaussianNB as GNB
nb_classifier = GNB()
nb_classifier.fit(X_train, y_train)
```

```
▾ GaussianNB
GaussianNB()
```

```
log_reg.fit(X_train, y_train)
```

```
▾         LogisticRegression
LogisticRegression(random_state=42)
```

```
dt.fit(X_train, y_train)
```

```
▾         DecisionTreeClassifier
DecisionTreeClassifier(random_state=42)
```

```
knn.fit(X_train, y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

```
voting_clf = VotingClassifier(
    estimators=[('dt', dt),('Naive Bayes',nb_classifier)],
    voting='hard' )

voting_clf.fit(X_train, y_train)
```

```
▸           VotingClassifier
          dt              Naive Bayes
▸ DecisionTreeClassifier  ▸ GaussianNB
```

```
voting_preds = voting_clf.predict(X_test)
voting_acc = accuracy_score(y_test, voting_preds)



print('Decision Tree accuracy:', accuracy_score(y_test, dt.predict(X_test)))
print('Naive Bayes accuracy:', accuracy_score(y_test, nb_classifier.predict(X_test)))
print('Ensemble accuracy:', voting_acc)
```

```
Decision Tree accuracy: 0.6458333333333334
Naive Bayes accuracy: 0.71875
Ensemble accuracy: 0.7135416666666666
```

```
from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression(random_state=42)
log_reg.fit(X_train, y_train)
y_pred_log_reg = log_reg.predict(X_test)

accuracy_log_reg = accuracy_score(y_test, y_pred_log_reg)
print("Logistic Regression Accuracy:", accuracy_log_reg)
```

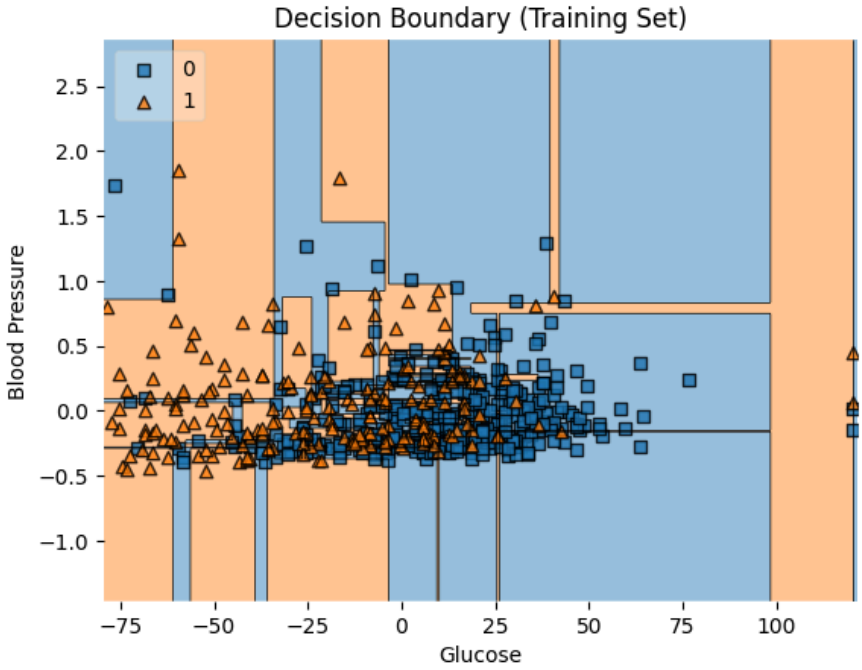
```
Logistic Regression Accuracy: 0.7395833333333334
```

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from mlxtend.plotting import plot_decision_regions

pca = PCA(n_components=2)
X_train_pca = pca.fit_transform(X_train)
X_test_pca = pca.transform(X_test)

clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train_pca, y_train)

plot_decision_regions(X_train_pca, y_train, clf=clf, legend=2)
plt.xlabel('Glucose')
plt.ylabel('Blood Pressure')
plt.title('Decision Boundary (Training Set)')
plt.show()
```

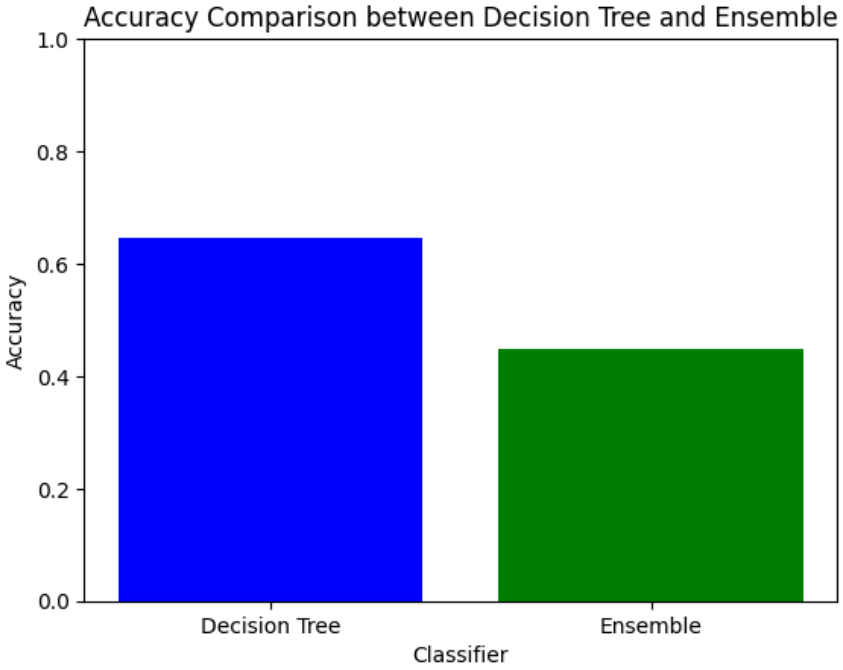



```
import matplotlib.pyplot as plt

# Calculate accuracies
dt_accuracy = accuracy_score(y_test, dt.predict(X_test))
ensemble_accuracy = accuracy_score(y_test, clf.predict(X_test))

# Plotting
classifiers = ['Decision Tree', 'Ensemble']
accuracies = [dt_accuracy, ensemble_accuracy]

plt.bar(classifiers, accuracies, color=['blue', 'green'])
plt.xlabel('Classifier')
plt.ylabel('Accuracy')
plt.title('Accuracy Comparison between Decision Tree and Ensemble')
plt.ylim(0, 1)  # Set y-axis limit to ensure proper visualization of accuracy values
plt.show()
```


Accuracy Comparison between Decision Tree and Ensemble
1.0
0.8
0.6
0.4
0.2
0.0
Accuracy
Decision Tree
Ensemble
Classifier