DecissionTreeClassifier como algoritmo de árboles de decision

```
from sklearn.tree import DecisionTreeClassifier

# Cargamos dataset
iris = load_iris()

#Dividimos entre entrenamiento y test
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, stratify=iris.target,
random_state=42)

#Creamos el modelo
tree = DecisionTreeClassifier(random_state=0)

#Entrenamos los datos
tree.fit(X_train, y_train)

#Obtenemos rendimiento sobre datos de entrenamiento y sobre datos de test
print("Accuracy on training set: {:.3f}".format(tree.score(X_train, y_train)))
print("Accuracy on test set: {:.3f}".format(tree.score(X_test, y_test)))
```

Podemos visualizar la importancia de cada característica de una manera que sea similar a la manera que visualizamos los coeficientes en el modelo linear

Importancia de características

Podemos valorar la importancia de cada característica para la decisión que toma un árbol.

```
def plot_feature_importances_iris(model):
        n_features = iris.data.shape[1]
        plt.barh(range(n_features), model.feature_importances_, align='center')
        plt.yticks(np.arange(n_features), iris.feature_names)
        plt.xlabel("Feature importance")
        plt.ylabel("Feature")

plot_feature_importances_iris(tree)
```

Visualización del árbol de decision

```
dot_data = StringIO.StringIO()
tree.export_graphviz(clf, out_file=dot_data,
feature_names=iris.feature_names,
class_names=iris.target_names,
filled=True, rounded=True,
special_characters=True)
graph = pydot.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```