|  |  |
| --- | --- |
| Título | |
| Código  import numpy as np  import matplotlib.pyplot as plt  import pandas as pd  dataset = pd.read\_csv('Social\_Network\_Ads.csv')  X = dataset.iloc[:, :-1].values  y= dataset.iloc[:, -1].values  # Splitting the dataset into the Training set and Test  from sklearn.model\_selection import train\_test\_split  X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25,random\_state= 0)  print(X\_train)  print(y\_train)  print(X\_test)  print(y\_test)  # Feature Scaling  from sklearn.preprocessing import StandardScaler  sc = StandardScaler()  X\_train = sc.fit\_transform(X\_train)  X\_test = sc.transform(X\_test)  print(X\_train)  print(X\_test)  from sklearn.svm import SVC  classifier = SVC(kernel = 'linear', random\_state = 0)  classifier.fit(X\_train, y\_train)  # Predicting a new result  print(classifier.predict(sc.transform([[30,87000]])))  #Predicting the Test set results  y\_pred = classifier.predict(X\_test)  print(np.concatenate((y\_pred.reshape(len(y\_pred),1),  y\_test.reshape(len(y\_test),1)),1))  #Making the Confusion Matrix  from sklearn.metrics import confusion\_matrix, accuracy\_score  cm = confusion\_matrix(y\_test, y\_pred)  print(cm)  accuracy\_score(y\_test, y\_pred)  # Visualising the Training set results  from matplotlib.colors import ListedColormap  X\_set, y\_set = sc.inverse\_transform(X\_train), y\_train  X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 10, stop = X\_set[:, 0].max() + 10, step = 0.25), np.arange(start = X\_set[:, 1].min() - 1000, stop = X\_set[:, 1].max() + 1000, step = 0.25)  plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(),X2.ravel()]).T)).reshape(X1.shape), alpha = 0.75, cmap = ListedColormap(('red','green')))  plt.xlim(X1.min(), X1.max())  plt.ylim(X2.min(), X2.max())  for i, j in enumerate(np.unique(y\_set)): plt.scatter(X\_set[y\_set ==j, 0], X\_set[y\_set == j, 1], c = ListedColormap(('red', 'green'))(i),label = j)  plt.title('SVM (Training set)')  plt.xlabel('Age')  plt.ylabel('Estimated Salary')  plt.legend()  plt.show() | |
| Ejecución | Código QR del repositorio en GitHub |