Day13

January 9, 2019

Finding the margin as big as possible

0.1 Data preprocessing

In [38]:
```
# import libraries
from sklearn.cross_validation import train_test_split
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline

# import dataset
dataset = pd.read_csv('../datasets/Social_Network_Ads.csv')
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# splitting data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.25, random_state=0)

# feature scaling
from sklearn.preprocessing import  StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.fit_transform(X_test)
```

```
/home/huiwen/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:475: DataConversi
  warnings.warn(msg, DataConversionWarning)
```

0.2 Fitting a SVM model

In [39]:
```
from sklearn.svm import SVC
model = SVC(kernel= 'linear', random_state=0)
model.fit(X_train, y_train)
```

Out[39]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='linear',
```

```
          max_iter=-1, probability=False, random_state=0, shrinking=True,
          tol=0.001, verbose=False)
```

0.3 Predict

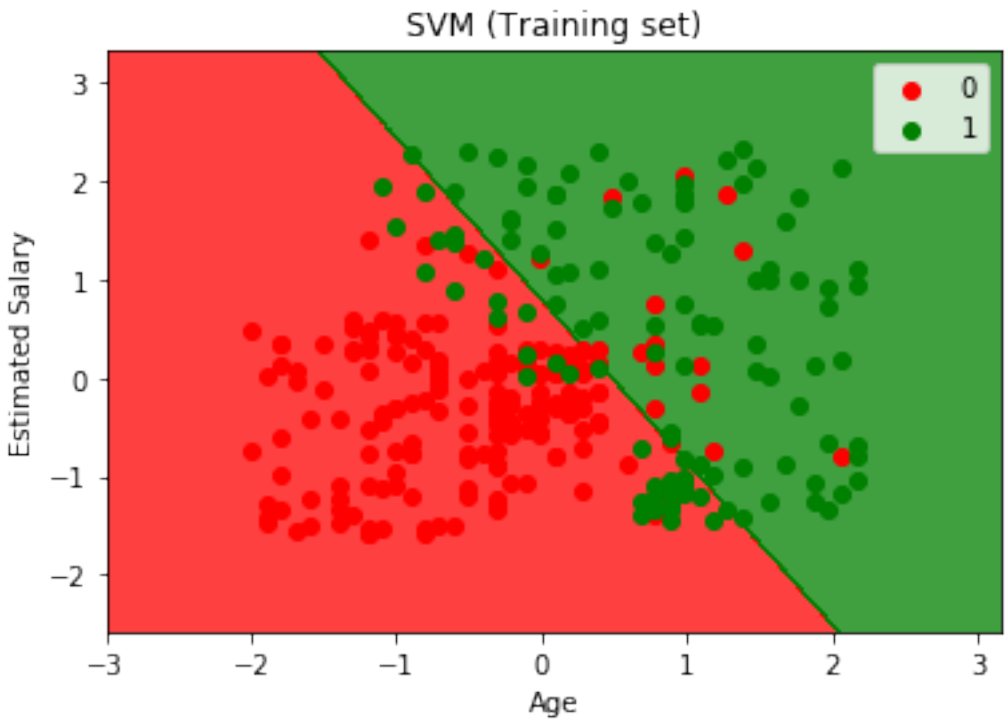
```
In [40]: y_pred = model.predict(X_test)
```

```
In [41]: from sklearn.metrics import confusion_matrix
         cm = confusion_matrix(y_test, y_pred)
```

0.4 Visualizing training set

- plot the boder
- plot the training data

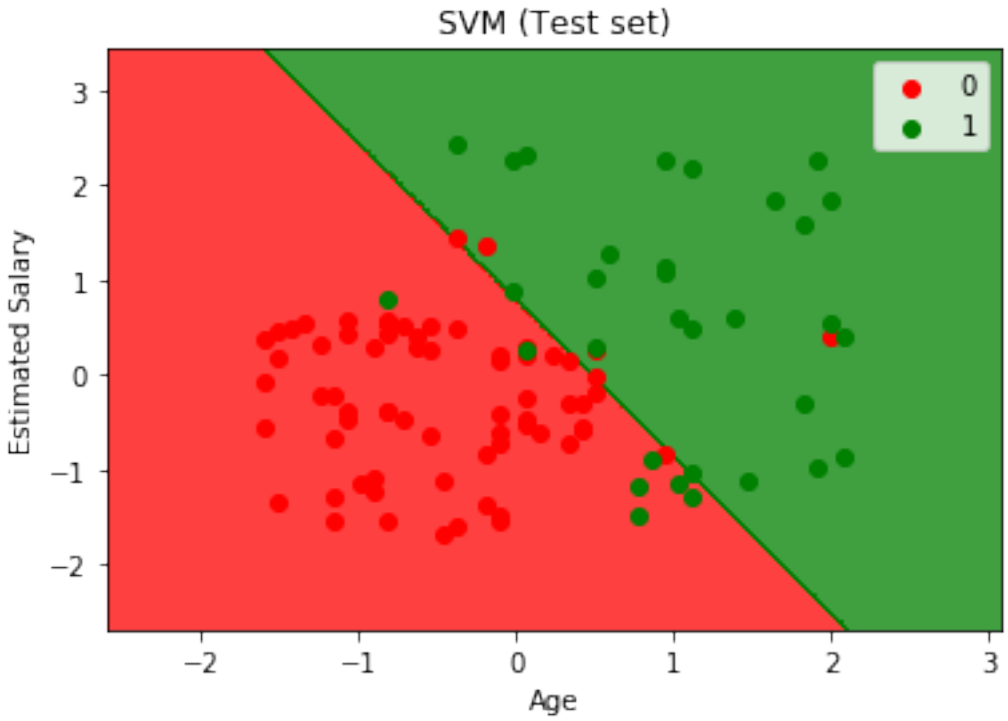
```
In [42]: from matplotlib.colors import ListedColormap
         X_set, y_set = X_train, y_train
         X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max()
                              np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max()
         # plot linear boder
         plt.contourf(X1, X2, model.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.sha
                      alpha = 0.75, cmap = ListedColormap(('red', 'green')))
         plt.xlim(X1.min(), X1.max())
         plt.ylim(X2.min(), X2.max())


         # plot training data
         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()
```



0.5 Visualizing test results

- plot border with test points

```
In [43]: from matplotlib.colors import ListedColormap
         X_set, y_set = X_test, y_test
         X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max()
                              np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max()
         plt.contourf(X1, X2, model.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.sha
                      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 (Test set)')
         plt.xlabel('Age')
         plt.ylabel('Estimated Salary')
         plt.legend()
         plt.show()
```


SVM (Test set)
Estimated Salary
Age
0
1
3
2
1
0
−1
−2
−2
−1
0
1
2
3