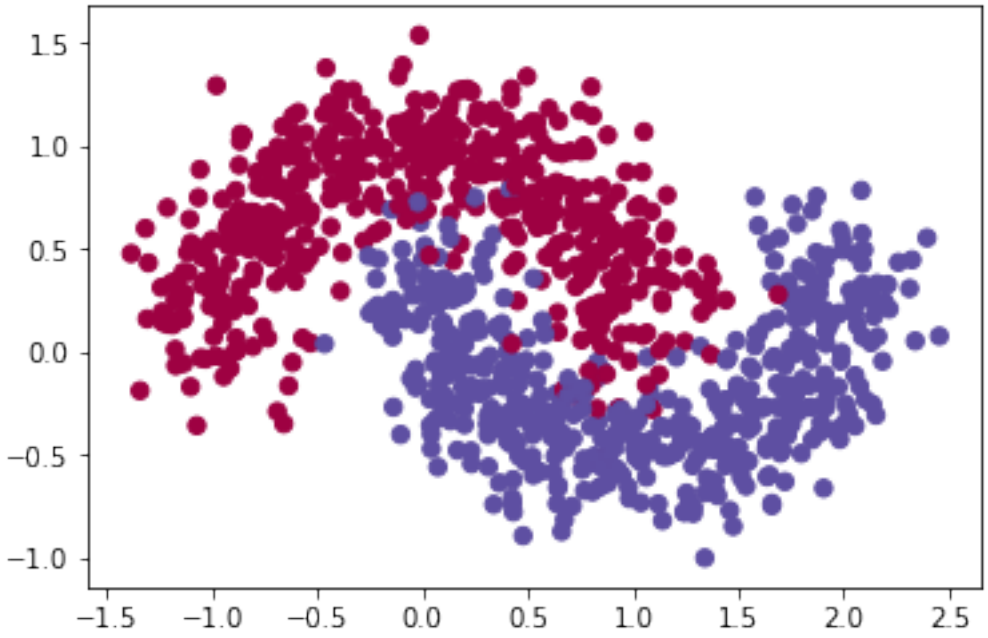
costobayesiano

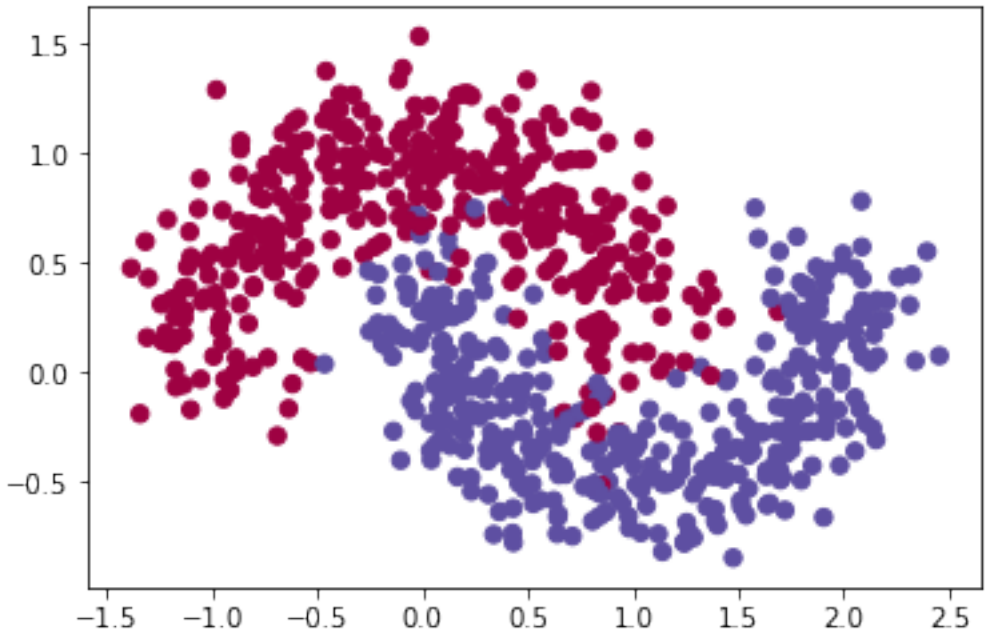
August 14, 2018

```
In [1]: import numpy as np

In [2]: import pandas as pd

In [3]: import matplotlib.pyplot as plt

In [4]: from sklearn.model_selection import cross_val_predict,StratifiedKFold,GridSearchCV,tra

In [5]: from sklearn.metrics import roc_auc_score

In [6]: from sklearn import tree

In [7]: from sklearn import datasets #Importamos el conjunto de datos

In [8]: from sklearn.model_selection import train_test_split

In [9]: np.random.seed(0)

In [10]: X, y = datasets.make_moons(1000, noise=0.20)

In [11]: #Dividimos nuestros datos en "conjunto de entrenamiento y de prueba

In [12]: X_train, X_test, y_train, y_test = train_test_split(X, y)

In [13]: plt.scatter(X[:,0], X[:,1], s=40, c=y, cmap=plt.cm.Spectral)

Out[13]: <matplotlib.collections.PathCollection at 0x1a0940c0d0>
```



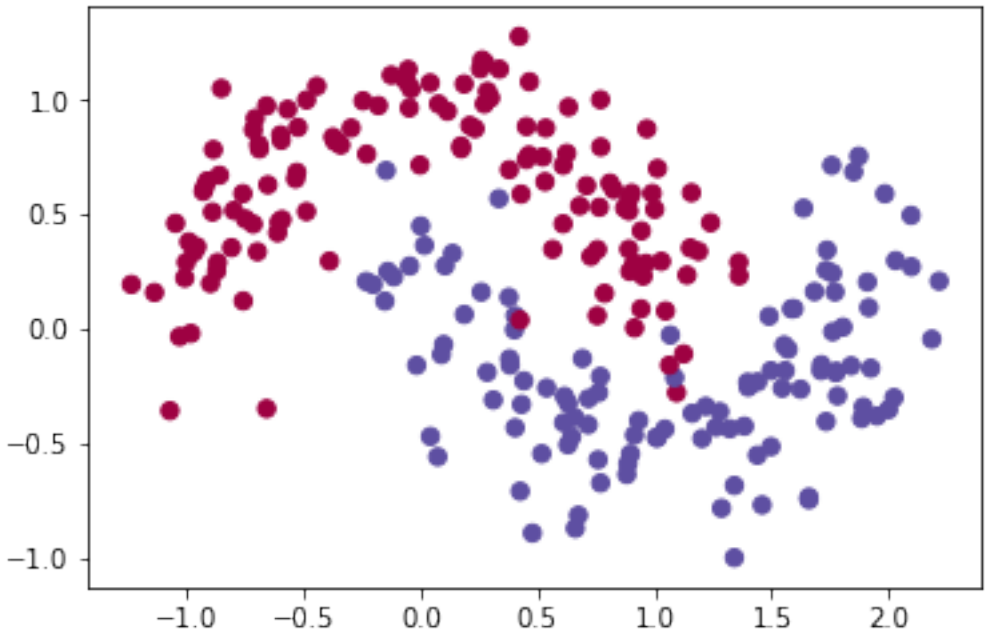

```
In [14]: plt.scatter(X_train[:,0], X_train[:,1], s=40, c=y_train, cmap=plt.cm.Spectral)
```

```
Out[14]: <matplotlib.collections.PathCollection at 0x1a0952b250>
```



In [15]:
```
plt.scatter(X_test[:,0], X_test[:,1], s=40, c=y_test, cmap=plt.cm.Spectral)
```

Out[15]: <matplotlib.collections.PathCollection at 0x1a096108d0>



In [118]:
```
#entreno un arbol como clasificador y como estimador de probabilidades

clf=tree.DecisionTreeClassifier(criterion='entropy',
                                min_samples_split=20,
                                min_samples_leaf=5,
                                max_depth = 10)
clf.fit(X_train,y_train)

# estimo probabilidades para cada muestra de test
proba=clf.predict_proba(X_test)

cost_01=10000
cost_00=0
cost_10=1
cost_11=0

#Calculo riesgos para cada decision

Riesgo1=proba[:,1]*cost_11+proba[:,0]*cost_10
Riesgo0=proba[:,1]*cost_01+proba[:,0]*cost_00

y_pred_test_bmr=(Riesgo0-Riesgo1)
```

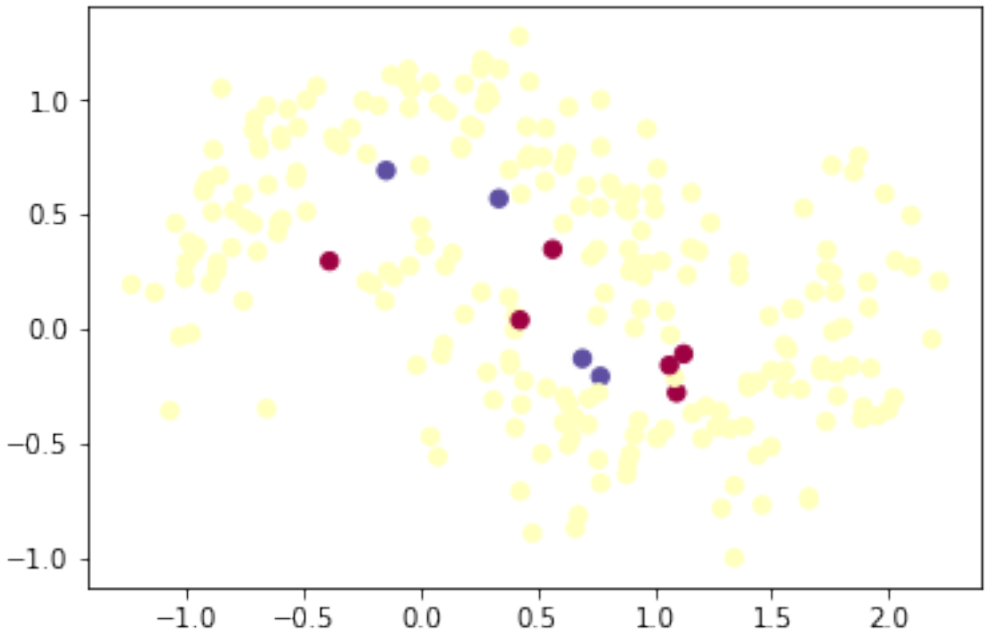
```
        #si el riesgo de asignar 0 es mayor que el de 1 decido 1

        y_pred_test_bmr1=(np.sign(y_pred_test_bmr)+1)/2
```

In [119]:
```
diff_test=y_test-clf.predict(X_test)
```

In [120]:
```
# resultados solo con el arbol
```

In [121]:
```
plt.scatter(X_test[:,0], X_test[:,1], s=40, c=diff_test, cmap=plt.cm.Spectral)
```

Out[121]:
```
<matplotlib.collections.PathCollection at 0x1a0adb5650>
```

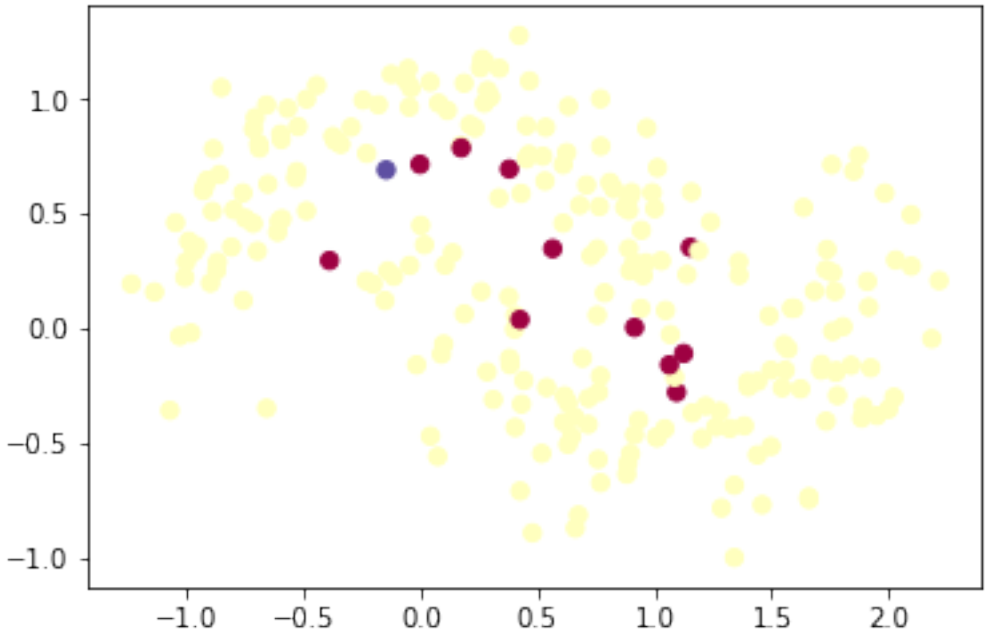


In [122]:
```
#resultado al incluir los costos de los errores
```

In [123]:
```
diff_test=y_test-y_pred_test_bmr1
```

In [124]:
```
plt.scatter(X_test[:,0], X_test[:,1], s=40, c=diff_test, cmap=plt.cm.Spectral)
```

Out[124]:
```
<matplotlib.collections.PathCollection at 0x1a0ae25f50>
```


1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0