1.2 Exercise: Retrain BDT with 2 features and plot decision boundary

Repeat the steps above but manually set the features to just two of the most “important” ones: `feature_names = ["DER_mass_MMC", "DER_mass_transverse_met_lep"]`

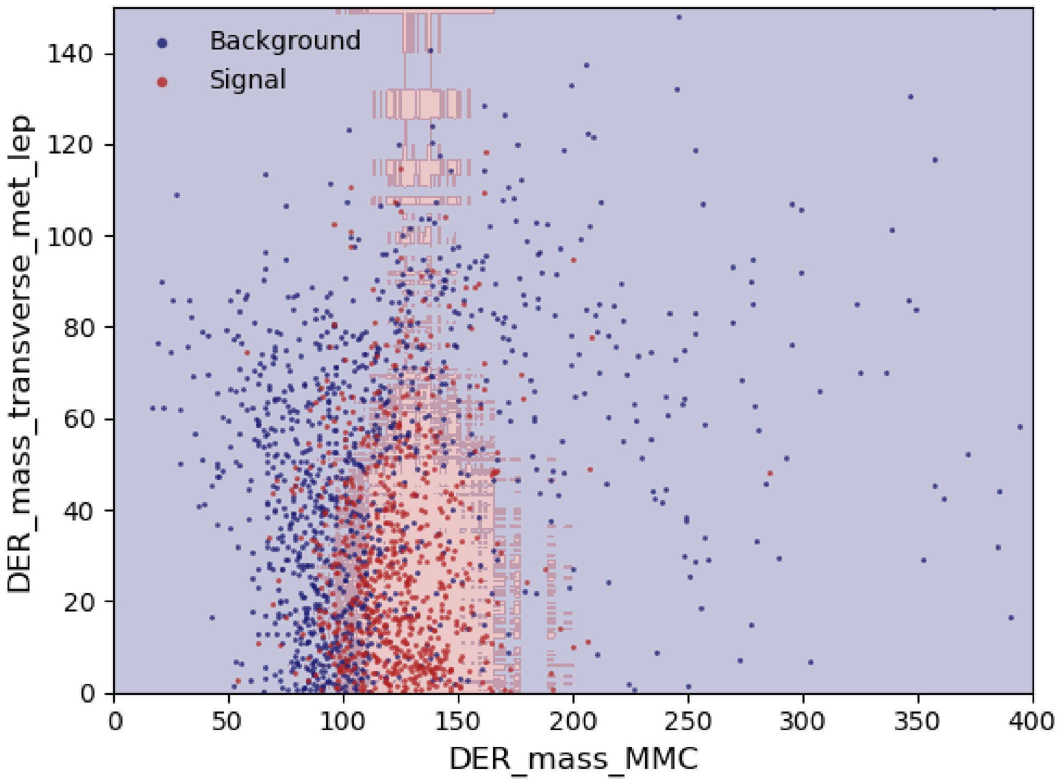
Then use the code below to plot the decision boundary in 2D. 1) What do you notice about the shape of the decision boundary? 2) Do you see any evidence of overfitting? How can you prove it? (Hint: consider the training data)

[20]:

```
feature_names = ["DER_mass_MMC", "DER_mass_transverse_met_lep"]

train = xgb.DMatrix(
    data=data_train[feature_names], label=data_train.Label.cat.codes,␣
 ↪missing=-999.0, feature_names=feature_names
)

test = xgb.DMatrix(
    data=data_test[feature_names], label=data_test.Label.cat.codes,␣
 ↪missing=-999.0, feature_names=feature_names
)

booster = xgb.train(param, train, num_boost_round=num_trees)
```

[21]:

```
from matplotlib.colors import ListedColormap

# first get a mesh grid
x_grid, y_grid = np.meshgrid(np.linspace(0, 400, 1000), np.linspace(0, 150,␣
 ↪1000))
# convert grid into DMatrix
matrix_grid = xgb.DMatrix(
    data=np.c_[x_grid.ravel(), y_grid.ravel()], missing=-999.0,␣
 ↪feature_names=feature_names
)
# run prediction for every value in grid
z_grid = booster.predict(matrix_grid)
# reshape
z_grid = z_grid.reshape(x_grid.shape)


plt.figure()
# plot decision boundary
ax = plt.subplot(111)
cm = ListedColormap(["midnightblue", "firebrick"])
plt.contourf(x_grid, y_grid, z_grid, levels=[0, 0.5, 1], cmap=cm, alpha=0.25)
# overlaid with test data points
plt.plot(
    DER_mass_MMC[mask_b],
    DER_mass_transverse_met_lep[mask_b],
    "o",
    markersize=2,
    color="midnightblue",
    markeredgewidth=0,
    alpha=0.8,
    label="Background",
)
plt.plot(
    DER_mass_MMC[mask_s],
    DER_mass_transverse_met_lep[mask_s],
    "o",
    markersize=2,
    color="firebrick",
    markeredgewidth=0,
    alpha=0.8,
    label="Signal",
)
ax.set_ylim(0,150)
ax.set_xlim(0,400)
plt.xlabel("DER_mass_MMC", fontsize=12)
plt.ylabel("DER_mass_transverse_met_lep", fontsize=12)
plt.legend(frameon=False, numpoints=1, markerscale=2)
plt.show()
```

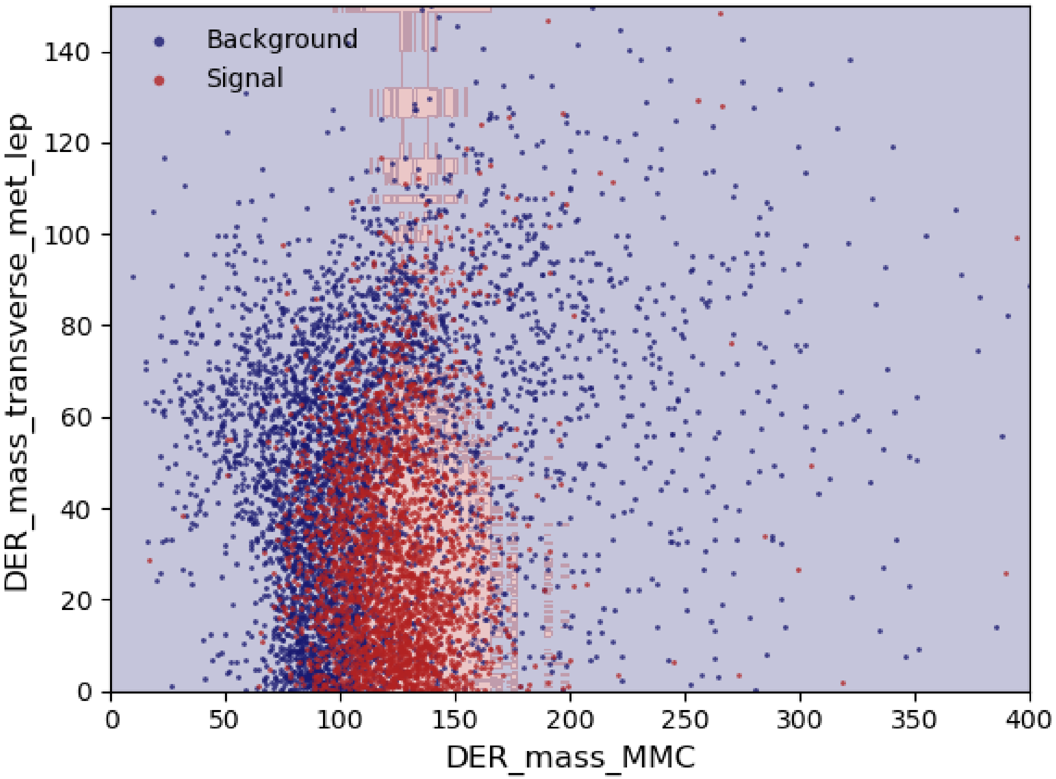


What I notice about the decision boundary is that it's made up of rectangles, not smooth at all. There is a concentrated mass around (x 100-150), (y 0-60). Which corresponds nicely with the 125 Gev mass of the Higgs Boson, fun fact.

I don't see real evidence of overfitting. Let's investigate a little:

[23]:

```
mask_b = np.array(data_train.Label == "b")
mask_s = np.array(data_train.Label == "s")

DER_mass_MMC = np.array(data_train.DER_mass_MMC)
DER_mass_transverse_met_lep = np.array(data_train.DER_mass_transverse_met_lep)

plt.figure()
# plot decision boundary
ax = plt.subplot(111)
cm = ListedColormap(["midnightblue", "firebrick"])
plt.contourf(x_grid, y_grid, z_grid, levels=[0, 0.5, 1], cmap=cm, alpha=0.25)
# overlaid with test data points
plt.plot(
    DER_mass_MMC[mask_b],
    DER_mass_transverse_met_lep[mask_b],
    "o",
    markersize=2,
    color="midnightblue",
    markeredgewidth=0,
    alpha=0.8,
    label="Background",
)
plt.plot(
    DER_mass_MMC[mask_s],
    DER_mass_transverse_met_lep[mask_s],
    "o",
    markersize=2,
    color="firebrick",
    markeredgewidth=0,
    alpha=0.8,
    label="Signal",
)
ax.set_ylim(0,150)
ax.set_xlim(0,400)
plt.xlabel("DER_mass_MMC", fontsize=12)
plt.ylabel("DER_mass_transverse_met_lep", fontsize=12)
plt.legend(frameon=False, numpoints=1, markerscale=2)
plt.show()
```



If there was widespread overfitting, I would expect to see a red dot in every rectangle. Looking at the plot with the training data instead of the test data, I can say that that is not the case. Therefore I proclaim: no overfitting!