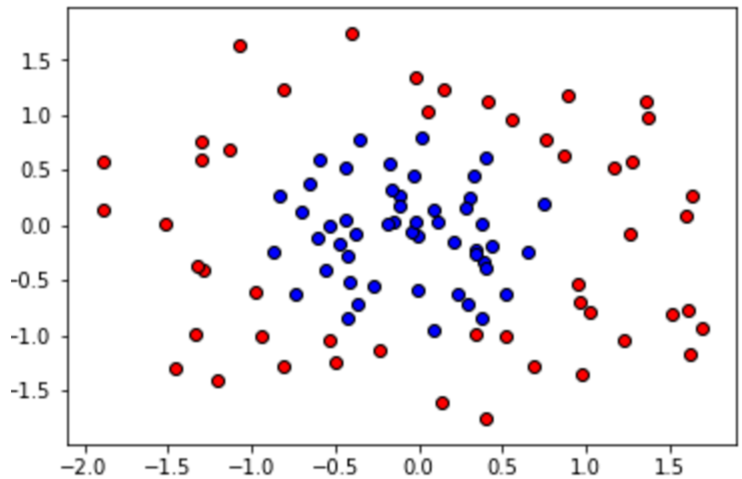
Detect Overfitting and Underfitting with Learning Curves

For this quiz, we'll be using three models to train the circular dataset below.

- A Decision Tree model,
- a Logistic Regression model, and
- a Support Vector Machine model.



One of the models overfits, one underfits, and the other one is just right. First, we'll write some code to draw the learning curves for each model, and finally we'll look at the learning curves to decide which model is which.

First, let's remember that the way the curves look for the three models, is as follows:

For the first part of the quiz, all you need is to uncomment one of the classifiers, and hit 'Test Run' to see the graph of the Learning Curve. But if you like coding, here are some details. We'll be using the function called `learning_curve`:

```
train_sizes, train_scores, test_scores = learning_curve(
    estimator, X, y, cv=None, n_jobs=1, train_sizes=np.linspace(.1, 1.0, num_trainings))
```

No need to worry about all the parameters of this function (you can read some more in here, but here we'll explain the main ones:

- `estimator`, is the actual classifier we're using for the data, e.g., `LogisticRegression()` or `GradientBoostingClassifier()`.
- `X` and `y` is our data, split into features and labels.
- `train_sizes` are the sizes of the chunks of data used to draw each point in the curve.
- `train_scores` are the training scores for the algorithm trained on each chunk of data.
- `test_scores` are the testing scores for the algorithm trained on each chunk of data.

Two very important observations:

- The training and testing scores come in as a list of 3 values, and this is because the function uses 3-Fold Cross-Validation.
- **Very important:** As you can see, we defined our curves with Training and Testing **Error**, and this function defines them with Training and Testing **Score**. These are

above.

Part 1: Drawing the learning curves

In here, we'll be comparing three models:

- A **Logistic Regression** model.
- A **Decision Tree** model.
- A **Support Vector Machine** model with an rbf kernel, and a gamma parameter of 1000 (we'll learn what these mean later).

Uncomment the code for each one, and examine the learning curve that gets drawn. If you're curious about the code used to draw the learning curves, it's on the **utils.py** tab.

quiz.py | utils.py | data.csv

```
# Import, read, and split data
import pandas as pd
data = pd.read_csv('data.csv')
import numpy as np
X = np.array(data[['x1', 'x2']])
y = np.array(data['y'])

# Fix random seed
np.random.seed(55)

### Imports
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.svm import SVC

# TODO: Uncomment one of the three classifiers, and hit "Test Run"
# to see the learning curve. Use these to answer the quiz below.

### Logistic Regression
#estimator = LogisticRegression()

### Decision Tree
#estimator = GradientBoostingClassifier()

### Support Vector Machine
estimator = SVC(kernel='rbf', gamma=1000)
```

Part 2: Analyzing the learning curves

For this second part of the quiz, you can look at the curves you've drawn before, to decide which one of the three models underfits, which one overfits, and which one is just right.

QUESTION 2 OF 2

From the models above, which model underfits, which one overfits, and which one is just right?

Submit to check your answer choices!

MODEL	OVERFITS, UNDEFITS, JUST RIGHT?
Logistic Regression	Underfits
Decision Tree	Just Right
Support Vector Machine	Overfit

SUBMIT

NEXT

Detecting Overfitting and Underfitting with Learning Curves