In [5]:
```
import pandas as pd
import numpy as np
```

In [6]:
```
from sklearn.datasets import fetch_openml
mnist = fetch_openml('mnist_784', version=1)
```

In [7]:
```
x,y = mnist["data"], mnist["target"]
x.shape
```

Out[7]:
```
(70000, 784)
```

In [8]:
```
y.shape
```

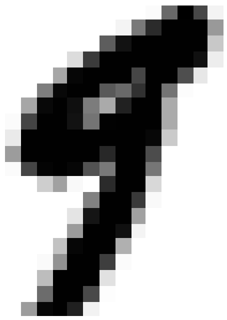
Out[8]:
```
(70000,)
```

To look into the datasamples:-

To view the image of a single digit,all we need to do is grab an instances feature vector,reshape it to 28x28 array,and display it using matplotlib's imshow() function.

In [9]:
```
%matplotlib inline
import matplotlib
import matplotlib.pyplot as plt
```

In [10]:
```
some_digit = x[36000] #selecting 36000th image in the dataset
some_digit_image = some_digit.reshape(28,28)#reshaping the image into 28x28 pixel
plt.imshow(some_digit_image, cmap=matplotlib.cm.binary, interpolation = "nearest'
plt.axis('off')
plt.show()
```



The image look like 9. Lets verify it.

In [11]:
```
y[18000]
```

Out[11]:
```
'4'
```

In [12]:

```
# Lets split the data into training and test with 60,000 images in training set

X_train, X_test, y_train, y_test = x[:60000], x[60000:], y[:60000], y[60000:]
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(60000, 784)
(10000, 784)
(60000,)
(10000,)
```

```
Also we need to shuffle our training data so that it ensures that we dont
missout any in the cross validation fold
```

In [13]:

```
import numpy as np
np.random.seed(42)
shuffle_index = np.random.permutation(60000)
X_train, y_train = X_train[shuffle_index], y_train[shuffle_index]
```

Forming the dataset and training the Classifier

In [14]:

```
#Example:-
from sklearn.linear_model import SGDClassifier
x1 = {"xcoord":pd.Series([0,1]),"ycoord":pd.Series([0,1])}
x3 = pd.DataFrame(x1)
y = [0,1]
clf = SGDClassifier(loss = "hinge", penalty = "l2")
clf.fit(x3,y)
#In SGDClassifier penalty is l2 not 12.
```

Out[14]:

```
SGDClassifier(alpha=0.0001, average=False, class_weight=None,
              early_stopping=False, epsilon=0.1, eta0=0.0, fit_intercept=True,
              l1_ratio=0.15, learning_rate='optimal', loss='hinge',
              max_iter=1000, n_iter_no_change=5, n_jobs=None, penalty='l2',
              power_t=0.5, random_state=None, shuffle=True, tol=0.001,
              validation_fraction=0.1, verbose=0, warm_start=False)
```

In [32]:

```
y_train_9 = (y_train == 9)
y_test_9 = (y_test == 9)
#y_train_9 = y_train_9.astype(np.uint8)
```

In [33]:

```
from sklearn.linear_model import SGDClassifier
sgd_clf = SGDClassifier(random_state=42, max_iter=10)
sgd_clf.fit(X_train, y_train_9)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-33-49c3c416df5b> in <module>
      1 from sklearn.linear_model import SGDClassifier
      2 sgd_clf = SGDClassifier(random_state=42, max_iter=10)
----> 3 sgd_clf.fit(X_train, y_train_9)

~\Anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py in f
it(self, X, y, coef_init, intercept_init, sample_weight)
    709                          loss=self.loss, learning_rate=self.learning_ra
te,
    710                          coef_init=coef_init, intercept_init=intercept_
init,
--> 711                          sample_weight=sample_weight)
    712
    713

~\Anaconda3\lib\site-packages\sklearn\linear_model\_stochastic_gradient.py in _
fit(self, X, y, alpha, C, loss, learning_rate, coef_init, intercept_init, sampl
e_weight)
    523
    524         X, y = check_X_y(X, y, 'csr', dtype=np.float64, order="C",
--> 525                          accept_large_sparse=False)
    526
    527         # labels can be encoded as float, int, or string literals

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_X_y(X, y, ac
cept_sparse, accept_large_sparse, dtype, order, copy, force_all_finite, ensure_
2d, allow_nd, multi_output, ensure_min_samples, ensure_min_features, y_numeric,
warn_on_dtype, estimator)
    758                         dtype=None)
    759     else:
--> 760         y = column_or_1d(y, warn=True)
    761         _assert_all_finite(y)
    762     if y_numeric and y.dtype.kind == 'O':

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in column_or_1d(y, wa
rn)
    795         return np.ravel(y)
    796
--> 797     raise ValueError("bad input shape {0}".format(shape))
    798
    799

ValueError: bad input shape ()
```

In [17]:

```
print(clf.coef_)
```

```
[[9.85221675 9.85221675]]
```

In [18]:
```
print(clf.intercept_[0])
```

```
-9.99002993014969
```

In [31]:
```
# Note that the label is a string. Most ML algorithms expect numbers, so let’s c
y_train = y_train_9.astype(np.uint8)
y_train = pd.to_numeric(y_train_9)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-31-d3962a4d6fb1> in <module>
      1 # Note that the label is a string. Most ML algorithms expect numbers, s
o let’s cast y to integer:
----> 2 y_train = y_train_9.astype(np.uint8)
      3 y_train = pd.to_numeric(y_train_9)

AttributeError: 'bool' object has no attribute 'astype'
```

In [20]:
```
y
```

Out[20]:
```
[0, 1]
```

In [21]:
```
# Lets split the data into training and test with 60,000 images in training set

X_train, X_test, y_train, y_test = x[:60000], x[60000:], y[:60000], y[60000:]
```

In [22]:
```
# Train the RandomForestClassifier

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_predict

forest_clf = RandomForestClassifier(random_state=42)
```

In [23]:

```
# Get Probabilities using cross_val_predict

y_probas_forest = cross_val_predict(forest_clf, X_train, y_train, cv=3, method="
y_probas_forest
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-23-ff122147d5fc> in <module>
      1 # Get Probabilities using cross_val_predict
      2 
----> 3 y_probas_forest = cross_val_predict(forest_clf, X_train, y_train, cv=3,
method="predict_proba")
      4 y_probas_forest

~\Anaconda3\lib\site-packages\sklearn\model_selection\_validation.py in cross_v
al_predict(estimator, X, y, groups, cv, n_jobs, verbose, fit_params, pre_dispat
ch, method)
    728     >>> y_pred = cross_val_predict(lasso, X, y, cv=3)
    729     """
--> 730     X, y, groups = indexable(X, y, groups)
    731 
    732     cv = check_cv(cv, y, classifier=is_classifier(estimator))

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in indexable(*iterabl
es)
    246     """
    247     result = [_make_indexable(X) for X in iterables]
--> 248     check_consistent_length(*result)
    249     return result
    250 

~\Anaconda3\lib\site-packages\sklearn\utils\validation.py in check_consistent_l
ength(*arrays)
    210     if len(uniques) > 1:
    211         raise ValueError("Found input variables with inconsistent numbe
rs of"
--> 212                          " samples: %r" % [int(l) for l in lengths])
    213 
    214 

ValueError: Found input variables with inconsistent numbers of samples: [60000,
2]
```

In [ ]: