Fetching Datasets

In [1]:
```
from sklearn.datasets import fetch_openml
```

In [2]:
```
mnist = fetch_openml('mnist_784')
```

In [3]:
```
mnist
```

Out[3]:
```
{'data': array([[0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        ...,
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.],
        [0., 0., 0., ..., 0., 0., 0.]]),
 'target': array(['5', '0', '4', ..., '4', '5', '6'], dtype=object),
 'frame': None,
 'feature_names': ['pixel1',
  'pixel2',
  'pixel3',
  'pixel4',
  'pixel5',
  'pixel6',
  'pixel7',
  'pixel8',
  'pixel9',
  'pixel10',
  'pixel11',
  'pixel12',
  'pixel13',
  'pixel14',
  'pixel15',
  'pixel16',
  'pixel17',
  'pixel18',
  'pixel19',
  'pixel20',
  'pixel21',
  'pixel22',
  'pixel23',
  'pixel24',
  'pixel25',
  'pixel26',
  'pixel27',
  'pixel28',
  'pixel29',
  'pixel30',
  'pixel31',
  'pixel32',
  'pixel33',
  'pixel34',
  'pixel35',
  'pixel36',
  'pixel37',
  'pixel38',
  'pixel39',
  'pixel40',
  'pixel41',
  'pixel42',
  'pixel43',
  'pixel44',
  'pixel45',
  'pixel46',
  'pixel47',
  'pixel48',
  'pixel49',
  'pixel50',
  'pixel51',
  'pixel52',
  'pixel53',
  'pixel54',
  'pixel55',
  'pixel56',
  'pixel57',
  'pixel58',
  'pixel59',
  'pixel60',
  'pixel61',
  'pixel62',
  'pixel63',
  'pixel64',
  'pixel65',
  'pixel66',
  'pixel67',
  'pixel68',
  'pixel69',
  'pixel70',
  'pixel71',
  'pixel72',
  'pixel73',
  'pixel74',
  'pixel75',
  'pixel76',
  'pixel77',
  'pixel78',
  'pixel79',
  'pixel80',
  'pixel81',
  'pixel82',
  'pixel83',
  'pixel84',
  'pixel85',
  'pixel86',
  'pixel87',
  'pixel88',
  'pixel89',
  'pixel90',
  'pixel91',
  'pixel92',
  'pixel93',
  'pixel94',
  'pixel95',
  'pixel96',
  'pixel97',
  'pixel98',
  'pixel99',
  'pixel100',
  'pixel101',
  'pixel102',
  'pixel103',
  'pixel104',
  'pixel105',
  'pixel106',
  'pixel107',
  'pixel108',
  'pixel109',
  'pixel110',
  'pixel111',
  'pixel112',
  'pixel113',
  'pixel114',
  'pixel115',
  'pixel116',
  'pixel117',
  'pixel118',
  'pixel119',
  'pixel120',
  'pixel121',
  'pixel122',
  'pixel123',
  'pixel124',
  'pixel125',
  'pixel126',
  'pixel127',
  'pixel128',
  'pixel129',
  'pixel130',
  'pixel131',
  'pixel132',
  'pixel133',
  'pixel134',
  'pixel135',
  'pixel136',
  'pixel137',
  'pixel138',
  'pixel139',
  'pixel140',
  'pixel141',
  'pixel142',
  'pixel143',
  'pixel144',
  'pixel145',
  'pixel146',
  'pixel147',
  'pixel148',
  'pixel149',
  'pixel150',
  'pixel151',
  'pixel152',
  'pixel153',
  'pixel154',
  'pixel155',
  'pixel156',
  'pixel157',
  'pixel158',
  'pixel159',
  'pixel160',
  'pixel161',
  'pixel162',
  'pixel163',
  'pixel164',
  'pixel165',
  'pixel166',
  'pixel167',
  'pixel168',
  'pixel169',
  'pixel170',
  'pixel171',
  'pixel172',
  'pixel173',
  'pixel174',
  'pixel175',
  'pixel176',
  'pixel177',
  'pixel178',
  'pixel179',
  'pixel180',
  'pixel181',
  'pixel182',
  'pixel183',
  'pixel184',
  'pixel185',
  'pixel186',
  'pixel187',
  'pixel188',
  'pixel189',
  'pixel190',
  'pixel191',
  'pixel192',
  'pixel193',
  'pixel194',
  'pixel195',
  'pixel196',
  'pixel197',
  'pixel198',
  'pixel199',
  'pixel200',
  'pixel201',
  'pixel202',
  'pixel203',
  'pixel204',
  'pixel205',
  'pixel206',
  'pixel207',
  'pixel208',
  'pixel209',
  'pixel210',
  'pixel211',
  'pixel212',
  'pixel213',
  'pixel214',
  'pixel215',
  'pixel216',
  'pixel217',
  'pixel218',
  'pixel219',
  'pixel220',
  'pixel221',
  'pixel222',
  'pixel223',
  'pixel224',
  'pixel225',
  'pixel226',
  'pixel227',
  'pixel228',
  'pixel229',
  'pixel230',
  'pixel231',
  'pixel232',
  'pixel233',
  'pixel234',
  'pixel235',
  'pixel236',
  'pixel237',
  'pixel238',
  'pixel239',
  'pixel240',
  'pixel241',
  'pixel242',
  'pixel243',
  'pixel244',
  'pixel245',
  'pixel246',
  'pixel247',
  'pixel248',
  'pixel249',
  'pixel250',
  'pixel251',
  'pixel252',
  'pixel253',
  'pixel254',
  'pixel255',
  'pixel256',
  'pixel257',
  'pixel258',
  'pixel259',
  'pixel260',
  'pixel261',
  'pixel262',
  'pixel263',
  'pixel264',
  'pixel265',
  'pixel266',
  'pixel267',
  'pixel268',
  'pixel269',
  'pixel270',
  'pixel271',
  'pixel272',
  'pixel273',
  'pixel274',
  'pixel275',
  'pixel276',
  'pixel277',
  'pixel278',
  'pixel279',
  'pixel280',
  'pixel281',
  'pixel282',
  'pixel283',
  'pixel284',
  'pixel285',
  'pixel286',
  'pixel287',
  'pixel288',
  'pixel289',
  'pixel290',
  'pixel291',
  'pixel292',
  'pixel293',
  'pixel294',
  'pixel295',
  'pixel296',
  'pixel297',
  'pixel298',
  'pixel299',
  'pixel300',
  'pixel301',
  'pixel302',
  'pixel303',
  'pixel304',
  'pixel305',
  'pixel306',
  'pixel307',
  'pixel308',
  'pixel309',
  'pixel310',
  'pixel311',
  'pixel312',
  'pixel313',
  'pixel314',
  'pixel315',
  'pixel316',
  'pixel317',
  'pixel318',
  'pixel319',
  'pixel320',
  'pixel321',
  'pixel322',
  'pixel323',
  'pixel324',
  'pixel325',
  'pixel326',
  'pixel327',
  'pixel328',
  'pixel329',
               'pixel330',
               'pixel331',
               'pixel332',
               'pixel333',
               'pixel334',
               'pixel335',
               'pixel336',
               'pixel337',
               'pixel338',
               'pixel339',
               'pixel340',
               'pixel341',
               'pixel342',
               'pixel343',
               'pixel344',
               'pixel345',
               'pixel346',
               'pixel347',
               'pixel348',
               'pixel349',
               'pixel350',
               'pixel351',
               'pixel352',
               'pixel353',
               'pixel354',
               'pixel355',
               'pixel356',
               'pixel357',
               'pixel358',
               'pixel359',
               'pixel360',
               'pixel361',
               'pixel362',
               'pixel363',
               'pixel364',
               'pixel365',
               'pixel366',
               'pixel367',
               'pixel368',
               'pixel369',
               'pixel370',
               'pixel371',
               'pixel372',
               'pixel373',
               'pixel374',
               'pixel375',
               'pixel376',
               'pixel377',
               'pixel378',
               'pixel379',
               'pixel380',
               'pixel381',
               'pixel382',
               'pixel383',
               'pixel384',
               'pixel385',
               'pixel386',
               'pixel387',
               'pixel388',
               'pixel389',
               'pixel390',
               'pixel391',
               'pixel392',
               'pixel393',
               'pixel394',
               'pixel395',
               'pixel396',
               'pixel397',
               'pixel398',
               'pixel399',
               'pixel400',
               'pixel401',
               'pixel402',
               'pixel403',
               'pixel404',
               'pixel405',
               'pixel406',
               'pixel407',
               'pixel408',
               'pixel409',
               'pixel410',
               'pixel411',
               'pixel412',
               'pixel413',
               'pixel414',
               'pixel415',
               'pixel416',
               'pixel417',
               'pixel418',
               'pixel419',
               'pixel420',
               'pixel421',
               'pixel422',
               'pixel423',
               'pixel424',
               'pixel425',
               'pixel426',
               'pixel427',
               'pixel428',
               'pixel429',
               'pixel430',
               'pixel431',
               'pixel432',
               'pixel433',
               'pixel434',
               'pixel435',
               'pixel436',
               'pixel437',
               'pixel438',
               'pixel439',
               'pixel440',
               'pixel441',
               'pixel442',
               'pixel443',
               'pixel444',
               'pixel445',
               'pixel446',
               'pixel447',
               'pixel448',
               'pixel449',
               'pixel450',
               'pixel451',
               'pixel452',
               'pixel453',
               'pixel454',
               'pixel455',
               'pixel456',
               'pixel457',
               'pixel458',
               'pixel459',
               'pixel460',
               'pixel461',
               'pixel462',
               'pixel463',
               'pixel464',
               'pixel465',
               'pixel466',
               'pixel467',
               'pixel468',
               'pixel469',
               'pixel470',
               'pixel471',
               'pixel472',
               'pixel473',
               'pixel474',
               'pixel475',
               'pixel476',
               'pixel477',
               'pixel478',
               'pixel479',
               'pixel480',
               'pixel481',
               'pixel482',
               'pixel483',
               'pixel484',
               'pixel485',
               'pixel486',
               'pixel487',
               'pixel488',
               'pixel489',
               'pixel490',
               'pixel491',
               'pixel492',
               'pixel493',
               'pixel494',
               'pixel495',
               'pixel496',
               'pixel497',
               'pixel498',
               'pixel499',
               'pixel500',
               'pixel501',
               'pixel502',
               'pixel503',
               'pixel504',
               'pixel505',
               'pixel506',
               'pixel507',
               'pixel508',
               'pixel509',
               'pixel510',
               'pixel511',
               'pixel512',
               'pixel513',
               'pixel514',
               'pixel515',
               'pixel516',
               'pixel517',
               'pixel518',
               'pixel519',
               'pixel520',
               'pixel521',
               'pixel522',
               'pixel523',
               'pixel524',
               'pixel525',
               'pixel526',
               'pixel527',
               'pixel528',
               'pixel529',
               'pixel530',
               'pixel531',
               'pixel532',
               'pixel533',
               'pixel534',
               'pixel535',
               'pixel536',
               'pixel537',
               'pixel538',
               'pixel539',
               'pixel540',
               'pixel541',
               'pixel542',
               'pixel543',
               'pixel544',
               'pixel545',
               'pixel546',
               'pixel547',
               'pixel548',
               'pixel549',
               'pixel550',
               'pixel551',
               'pixel552',
               'pixel553',
               'pixel554',
               'pixel555',
               'pixel556',
               'pixel557',
               'pixel558',
               'pixel559',
               'pixel560',
               'pixel561',
               'pixel562',
               'pixel563',
               'pixel564',
               'pixel565',
               'pixel566',
               'pixel567',
               'pixel568',
               'pixel569',
               'pixel570',
               'pixel571',
               'pixel572',
               'pixel573',
               'pixel574',
               'pixel575',
               'pixel576',
               'pixel577',
               'pixel578',
               'pixel579',
               'pixel580',
               'pixel581',
               'pixel582',
               'pixel583',
               'pixel584',
               'pixel585',
               'pixel586',
               'pixel587',
               'pixel588',
               'pixel589',
               'pixel590',
               'pixel591',
               'pixel592',
               'pixel593',
               'pixel594',
               'pixel595',
               'pixel596',
               'pixel597',
               'pixel598',
               'pixel599',
               'pixel600',
               'pixel601',
               'pixel602',
               'pixel603',
               'pixel604',
               'pixel605',
               'pixel606',
               'pixel607',
               'pixel608',
               'pixel609',
               'pixel610',
               'pixel611',
               'pixel612',
               'pixel613',
               'pixel614',
               'pixel615',
               'pixel616',
               'pixel617',
               'pixel618',
               'pixel619',
               'pixel620',
               'pixel621',
               'pixel622',
               'pixel623',
               'pixel624',
               'pixel625',
               'pixel626',
               'pixel627',
               'pixel628',
               'pixel629',
               'pixel630',
               'pixel631',
               'pixel632',
               'pixel633',
               'pixel634',
               'pixel635',
               'pixel636',
               'pixel637',
               'pixel638',
               'pixel639',
               'pixel640',
               'pixel641',
               'pixel642',
               'pixel643',
               'pixel644',
               'pixel645',
               'pixel646',
               'pixel647',
               'pixel648',
               'pixel649',
               'pixel650',
               'pixel651',
               'pixel652',
               'pixel653',
               'pixel654',
               'pixel655',
               'pixel656',
               'pixel657',
               'pixel658',
               'pixel659',
               'pixel660',
               'pixel661',
               'pixel662',
               'pixel663',
               'pixel664',
               'pixel665',
               'pixel666',
               'pixel667',
               'pixel668',
               'pixel669',
               'pixel670',
               'pixel671',
               'pixel672',
               'pixel673',
               'pixel674',
               'pixel675',
               'pixel676',
               'pixel677',
               'pixel678',
               'pixel679',
               'pixel680',
               'pixel681',
               'pixel682',
               'pixel683',
               'pixel684',
               'pixel685',
               'pixel686',
               'pixel687',
               'pixel688',
               'pixel689',
               'pixel690',
               'pixel691',
               'pixel692',
               'pixel693',
               'pixel694',
               'pixel695',
               'pixel696',
               'pixel697',
               'pixel698',
               'pixel699',
               'pixel700',
               'pixel701',
               'pixel702',
               'pixel703',
               'pixel704',
               'pixel705',
               'pixel706',
               'pixel707',
               'pixel708',
               'pixel709',
               'pixel710',
               'pixel711',
               'pixel712',
               'pixel713',
               'pixel714',
               'pixel715',
               'pixel716',
               'pixel717',
               'pixel718',
               'pixel719',
               'pixel720',
               'pixel721',
               'pixel722',
               'pixel723',
               'pixel724',
               'pixel725',
               'pixel726',
               'pixel727',
               'pixel728',
               'pixel729',
               'pixel730',
               'pixel731',
               'pixel732',
               'pixel733',
               'pixel734',
               'pixel735',
               'pixel736',
               'pixel737',
               'pixel738',
               'pixel739',
               'pixel740',
               'pixel741',
               'pixel742',
               'pixel743',
               'pixel744',
               'pixel745',
               'pixel746',
               'pixel747',
               'pixel748',
               'pixel749',
               'pixel750',
               'pixel751',
               'pixel752',
               'pixel753',
               'pixel754',
               'pixel755',
               'pixel756',
               'pixel757',
               'pixel758',
               'pixel759',
               'pixel760',
               'pixel761',
               'pixel762',
               'pixel763',
               'pixel764',
               'pixel765',
               'pixel766',
               'pixel767',
               'pixel768',
               'pixel769',
               'pixel770',
               'pixel771',
               'pixel772',
               'pixel773',
               'pixel774',
               'pixel775',
               'pixel776',
               'pixel777',
               'pixel778',
               'pixel779',
               'pixel780',
               'pixel781',
               'pixel782',
               'pixel783',
               'pixel784'],
              'target_names': ['class'],
              'DESCR': "**Author**: Yann LeCun, Corinna Cortes, Christopher J.C. Bur
             ges  \n**Source**: [MNIST Website](http://yann.lecun.com/exdb/mnist/) -
             Date unknown  \n**Please cite**:  \n\nThe MNIST database of handwritten
             digits with 784 features, raw data available at: http://yann.lecun.com/
             exdb/mnist/. It can be split in a training set of the first 60,000 exam
             ples, and a test set of 10,000 examples  \n\nIt is a subset of a larger
             set available from NIST. The digits have been size-normalized and cente
             red in a fixed-size image. It is a good database for people who want to
             try learning techniques and pattern recognition methods on real-world d
             ata while spending minimal efforts on preprocessing and formatting. The
             original black and white (bilevel) images from NIST were size normalize
             d to fit in a 20x20 pixel box while preserving their aspect ratio. The
```

```
resulting images contain grey levels as a result of the anti-aliasing t
echnique used by the normalization algorithm. the images were centered
in a 28x28 image by computing the center of mass of the pixels, and tra
nslating the image so as to position this point at the center of the 28
x28 field.  \n\nWith some classification methods (particularly template
-based methods, such as SVM and K-nearest neighbors), the error rate im
proves when the digits are centered by bounding box rather than center
of mass. If you do this kind of pre-processing, you should report it in
your publications. The MNIST database was constructed from NIST's NIST
originally designated SD-3 as their training set and SD-1 as their test
set. However, SD-3 is much cleaner and easier to recognize than SD-1. T
he reason for this can be found on the fact that SD-3 was collected amo
ng Census Bureau employees, while SD-1 was collected among high-school
students. Drawing sensible conclusions from learning experiments requir
es that the result be independent of the choice of training set and tes
t among the complete set of samples. Therefore it was necessary to buil
d a new database by mixing NIST's datasets.  \n\nThe MNIST training set
is composed of 30,000 patterns from SD-3 and 30,000 patterns from SD-1.
Our test set was composed of 5,000 patterns from SD-3 and 5,000 pattern
s from SD-1. The 60,000 pattern training set contained examples from ap
proximately 250 writers. We made sure that the sets of writers of the t
raining set and test set were disjoint. SD-1 contains 58,527 digit imag
es written by 500 different writers. In contrast to SD-3, where blocks
of data from each writer appeared in sequence, the data in SD-1 is scra
mbled. Writer identities for SD-1 is available and we used this informa
tion to unscramble the writers. We then split SD-1 in two: characters w
ritten by the first 250 writers went into our new training set. The rem
aining 250 writers were placed in our test set. Thus we had two sets wi
th nearly 30,000 examples each. The new training set was completed with
enough examples from SD-3, starting at pattern # 0, to make a full set
of 60,000 training patterns. Similarly, the new test set was completed
with SD-3 examples starting at pattern # 35,000 to make a full set with
60,000 test patterns. Only a subset of 10,000 test images (5,000 from S
D-1 and 5,000 from SD-3) is available on this site. The full 60,000 sam
ple training set is available.\n\nDownloaded from openml.org.",
 'details': {'id': '554',
  'name': 'mnist_784',
  'version': '1',
  'format': 'ARFF',
  'upload_date': '2014-09-29T03:28:38',
  'licence': 'Public',
  'url': 'https://www.openml.org/data/v1/download/52667/mnist_784.arf
f',
  'file_id': '52667',
  'default_target_attribute': 'class',
  'tag': ['AzurePilot',
   'OpenML-CC18',
   'OpenML100',
   'study_1',
   'study_123',
   'study_41',
   'study_99',
   'vision'],
  'visibility': 'public',
  'status': 'active',
  'processing_date': '2018-10-03 21:23:30',
  'md5_checksum': '0298d579eb1b86163de7723944c7e495'},
 'categories': {},
 'url': 'https://www.openml.org/d/554'}
```

In [4]:
```
x, y = mnist['data'], mnist['target']
```

In [5]:
```
x
```

Out[5]:
```
array([[0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       ...,
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.],
       [0., 0., 0., ..., 0., 0., 0.]])
```

In [6]:
```
y
```

Out[6]:
```
array(['5', '0', '4', ..., '4', '5', '6'], dtype=object)
```

In [7]:
```
x[0]
```

```
Out[7]: array([  0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    3.,   1
        8.,
                18.,   18.,  126.,  136.,  175.,   26.,  166.,  255.,  247.,  127.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                30.,   36.,   94.,  154.,  170.,  253.,  253.,  253.,  253.,  253.,  22
        5.,
               172.,  253.,  242.,  195.,   64.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,   49.,  238.,  253.,  253.,  253.,  25
        3.,
               253.,  253.,  253.,  253.,  251.,   93.,   82.,   82.,   56.,   39.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                18., 219., 253., 253., 253., 253., 253., 198., 182., 247., 24
        1.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,  80., 156., 107., 25
        3.,
               253., 205.,  11.,   0.,  43., 154.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,  14.,   1., 154., 253.,  90.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
               139., 253., 190.,   2.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,  11., 190., 253.,  70.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,  35., 241., 225., 160., 108.,   1.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,  81., 24
        0.,
               253., 253., 119.,  25.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,   45., 186., 253., 253., 150.,   2
        7.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,   16.,   93., 252., 253., 187.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0., 249., 25
        3.,
               249.,   64.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,   46., 130., 183., 253., 253., 207.,    2.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,   39., 148., 229., 253., 253., 25
        3.,
               250., 182.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,   24., 11
        4.,
               221., 253., 253., 253., 253., 201.,   78.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,   23.,   66., 213., 253., 253., 253., 253., 198.,   8
        1.,
                 2.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,   18., 171., 219., 253., 25
        3.,
               253., 253., 195.,   80.,    9.,    0.,    0.,    0.,    0.,    0.,
        0.,
                 0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,    0.,   5
```

```
5.,
        172., 226., 253., 253., 253., 253., 244., 133.,  11.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0., 136., 253., 253., 253., 212., 13
5.,
        132.,  16.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,   0.,
0.,
          0.,   0.,   0.])
```

In [8]:
```
len(x)
```

Out[8]:
```
70000
```

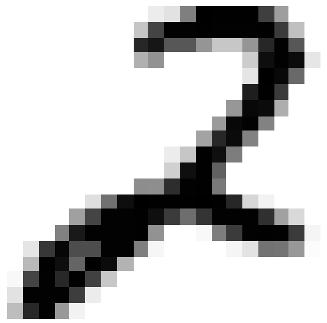
In [9]:
```
len(x[0])
```

Out[9]:
```
784
```

In [10]:
```
x.shape
```

Out[10]:
```
(70000, 784)
```

In [11]:
```
x[0].shape
```
Out[11]:
```
(784,)
```
In [12]:
```
len(y)
```
Out[12]:
```
70000
```
In [13]:
```
y.shape
```
Out[13]:
```
(70000,)
```
In [14]:
```
%matplotlib inline
```
In [15]:
```
import matplotlib
import matplotlib.pyplot as plt
```
In [16]:
```
some_digit = x[36001]
some_digit_image = some_digit.reshape(28, 28)
```
In [17]:
```
plt.imshow(some_digit_image, cmap = matplotlib.cm.binary, interpolation
 = 'nearest')
plt.axis('off')
```
Out[17]:
```
(-0.5, 27.5, 27.5, -0.5)
```



```
In [18]: y[36001]
```

```
Out[18]: '2'
```

```
In [19]: x_train, x_test = x[:60000], x[60000:]
         y_train, y_test = y[:60000], y[60000:]
```

```
In [20]: import numpy as np
         np.random.seed(42)
         shuffle_index = np.random.permutation(70000)
         x_train, y_train = x[shuffle_index[:60000]], y[shuffle_index[:60000]]
         x_test, y_test = x[shuffle_index[60000:]], y[shuffle_index[60000:]]
```

Creating a Two Detector

```
In [46]: y_train = y_train.astype(np.int8)
         y_test = y_test.astype(np.int8)
         y_train_2 = (y_train == 2)
         y_test_2 = (y_test == 2)
```

In [47]:
```
y_train
```

Out[47]:
```
array([8, 4, 8, ..., 0, 8, 1], dtype=int8)
```

In [48]:
```
y_train_2
```

Out[48]:
```
array([False, False, False, ..., False, False, False])
```

In [49]:
```
y_test_2
```

Out[49]:
```
array([False, False, False, ..., False, False, False])
```

In [50]:
```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression()
```

In [51]:
```
clf.fit(x_train, y_train_2)
```

```
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[51]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=
True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbo
se=0,
                   warm_start=False)
```

In [52]:
```
clf.predict([some_digit])
```

Out[52]:
```
array([ True])
```

In [53]:
```
from sklearn.model_selection import cross_val_score
a = cross_val_score(clf, x_train, y_train_2, cv = 3, scoring = 'accurac
y')
```

```
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
```

```
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

```
In [54]: a
```

```
Out[54]: array([0.97865, 0.9784 , 0.97925])
```

```
In [55]: a.mean()
```

```
Out[55]: 0.9787666666666667
```

```
In [56]: from sklearn.model_selection import cross_val_predict
         y_train_pred = cross_val_predict(clf, x_train, y_train_2, cv = 3)
```

```
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
```

```
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

In [57]:
```
y_train_pred
```

Out[57]:
```
array([False, False, False, ..., False, False, False])
```

Calculating Confusion Matrix

In [58]:
```
from sklearn.metrics import confusion_matrix
```

In [59]:
```
confusion_matrix(y_train_2, y_train_pred)
```

Out[59]:
```
array([[53575,   465],
       [  809,  5151]], dtype=int64)
```

In [60]:
```
confusion_matrix(y_train_2, y_train_2)
```

Out[60]:
```
array([[54040,     0],
       [    0,  5960]], dtype=int64)
```

Precision And Recall

```
In [61]: from sklearn.metrics import precision_score, recall_score
```

```
In [62]: precision_score(y_train_2, y_train_pred)

Out[62]: 0.9172008547008547
```

```
In [63]: recall_score(y_train_2, y_train_pred)

Out[63]: 0.8642617449664429
```

F-1 Score

```
In [64]: from sklearn.metrics import f1_score
```

```
In [65]: f1_score(y_train_2, y_train_pred)

Out[65]: 0.8899447131997236
```

Precision Recall Curve

```
In [66]: from sklearn.metrics import precision_recall_curve
```

```
In [79]: y_scores = cross_val_predict(clf, x_train, y_train_2, method = 'decisio
         n_function')
```

```
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
```

```
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
c:\users\user\appdata\local\programs\python\python37\lib\site-packages
```

```
\sklearn\linear_model\_logistic.py:940: ConvergenceWarning: lbfgs faile
d to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown
in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

In [84]:
```
precisions, recalls, thresholds = precision_recall_curve(y_train_2, y_s
cores)
```

In [85]:
```
precision
```

Out[85]:
```
array([0.09933333, 0.91720085, 1.        ])
```

In [86]:
```
recalls
```

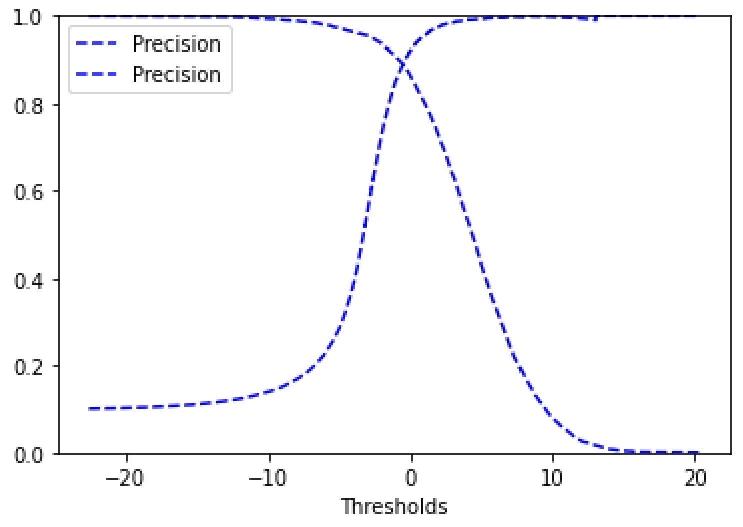
Out[86]:
```
array([1.00000000e+00, 9.99832215e-01, 9.99832215e-01, ...,
       3.35570470e-04, 1.67785235e-04, 0.00000000e+00])
```

In [87]:
```
thresholds
```

Out[87]:
```
array([-22.59776586, -22.59761139, -22.59133787, ...,  18.07776875,
        18.729092  ,  20.34568722])
```

Plotting The Precision Recall Curve

In [90]:
```
plt.plot(thresholds, precisions[:-1], "b--", label = "Precision")
plt.plot(thresholds, recalls[:-1], "b--", label = "Rec" )
plt.xlabel("Thresholds")
plt.legend(loc = 'upper left')
plt.ylim([0, 1])
plt.show()
```



```
In [ ]:
```