28|3|24

Hands on Machine Learning with Scikit Learn, keras and Tensor flow.

1. Get the Data

Download
the
Data

```
import os
import tarlite
import urllib
def fetch_housing data (housing_url =
                Housing_URL housing-path = Housing PATH):
urllib.request.urlretrieve (url, housing/url)"
  housing_tag = tatife.open(name = tag_Path)
   housing_tag 2. extract all(path = housing_path)
   housing_tag 2. close ()
     fetch_housing_data()
```

Load
the
data

```
import Pandas as pd
def load_housing data (housing_path = HOUSINGPATH)
    data_path = os.path.join
     return pd.read_csv (data_path)
         housing.head ()
          housing.info ()
housing (ocean + Proximity) value, count)
           housing, distribe ()
```

create
a
Test
set

```
def split_train.test (data, test_ratio =0.2)
Shuffled.indicies = np.random permutation
  test_set_size = int (len(data) * test_ratio)
  test_indices = shuffled.indices (test_set_size)
     return data loc (train index)

      def test_set_check (indentifier, test_ratio =2)
           total_size = 2x *32
       intrust_hex_vpr_2 (test_ratio * total size)
```

```
    dif cantor_Pailing (n1, n2)
    n = ((n1 + n2) * (n1 + n2 + 1) ) / 2 ) + n2
        return n


  dif  lat_lon_to_index (lat, lon)
    lat, lon = int (lat x 100) , int (lon x 100)
    lat, lon = from _ z _ to _ N(lat), from z to N(lon)
    index = contor_ Pairing (lat, lon)
        return np. int64 (index)


    dif from    z to N(z)
            if z >= 0
              n = 2 x z ;
        else
           n = -2 x z - 1
              return n
```

(2) Discover and visualize the Data to gain insight

```
Start_train_set. shape ., etrat_test _set _shape
Start_ test_set . rest. index () to framer
                                   (frame =' data 101 /
 hovvry = Start_train set copy()            start_test.
```

visualizing geographical Data

```
housing.plot ( land = 'scatter', x = longitude , y = latitude
          Plot_show ()

housing_plot (land = 'Scatter; x = longitude up to = 0, v
          plt.show()
```

3 Prepare the Data for machine learning Algorithm

```
housing = start train . set. drop ("median _ house . value_
                                    axis = 1 )
housing _ labels = start. train _ set ("median _ house_
                                       value ]
                                          copy ()
housing shape, housing _labels shape
```

→ Data cleaning

```
from sklearn impute import simple imputer
 imputer = simple Imputer (strategy = 'median' )
      imputor - fit ( housing - num )
         imputor . statistics
              x shape
     housing - num median () values
       housing - (head )
```

→ custom transformer

```
from sklearn . base import Transformer m,
                                     Base estimator
class combined Attribute Adder ( Base _ estimator,
                                    Transformer mixin)
  self . add _ bedrooms - per _ room =
               add -bedroom perroom
def fit ( sort, x, y, = none)
    rooms per - household = 4 [:, rooms _ (1] /
                           x [: , household - (1 ]
housing extra_attribs = attr_adder.transform
                            (housing values),
```

→ transformation pipelines.

```
from sklearn.pipeline import pipeline
from sklearn.preprocessing import standard scaler
num = pipeline = pipelin
  (imputer = simple Imputer (stratergy = 'median'))
```

4 Select and train a Model.

```
from sklearn.linear_model import linear Regression
 lin_reg = linear Regression ()
  Some_data = housing.iloc [:5]
   Some_data_prepared = full_pipeline.transform
                              (some_data)
```

5 Fine Tune your model.

Grid search

```
from sklearn.model_selection import Gridsearch
  { n_estimator [3,10,30]
   { bootstrap (false) , n_estimator (3,10),
      max_feature (2,3,4) ,
    },
   forest_reg = Random forest Research ()
      guide_search_best_param.
```

6 Launch, monitor and maintain your system.