**Name : Adham mohamed labeb**

**Level : 3**

# \* Pattern Recognition Steps :-

1. Train:-
   * Training data .
   * Pre processing of the photos .
   * Feature extraction .
   * Features.
   * Model Learning .
2. Test :-
   * + Physical Enivronment .
     + Bring data I am working on to code (Data Acquisition ).
     + Pre Proccesing :- to process the data from any noise .
     + Feature Extraction :- Extract the features to (discriminative &invariant features ) , turning them into vectors .
     + Classification (assign the objects to categories ) .

-post processing

* + - Decision

# \*Types Of Classifications :-

**-**



(Depends on probability , takes less time in training but more in testing , class conditional independence) ex:NB classifier .

**-**Discreminative :-

(Depends on ML , take less time in testing more in training , class conditional dependence ) ex (RF , ANN,LR)classifiers .

# \*Measuring accuracy , precision and the other by 2 methods :-

- Holdout (Random )

-Cross Validation



2

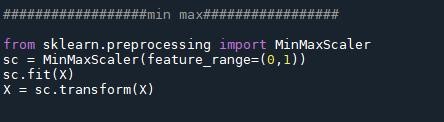
- Data Acquisition:-

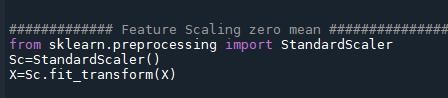


It is to get the data (photos ) and work on it ( fitting photos size by (256\*256)by reshaping it , transform it into values and get it in 2 arrays , X,Y ) .

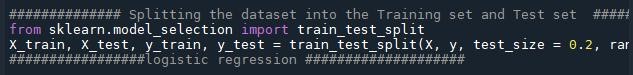
-Normalization Data :-

2 ways (min-max or z score )

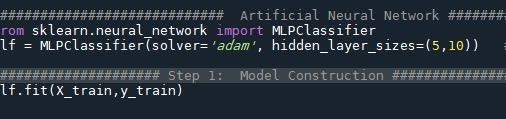




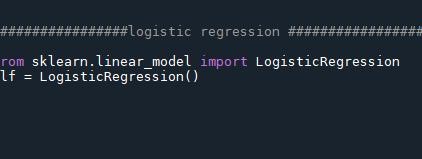
-Split the data into train , test data :-



- Calling the needed Algorithm (LR ,RF,ANN,NB)



Artificial Neural Network



## Logistic Regression

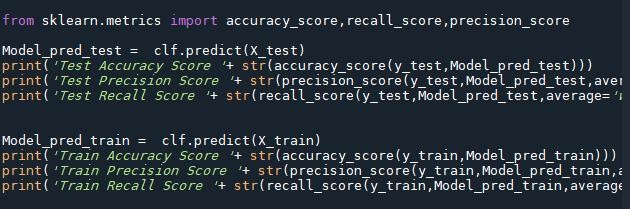


## Naive Baise

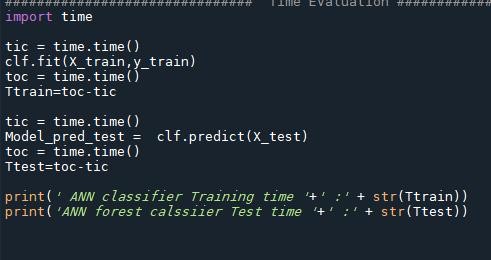


Random Forest

-Get the accuracy of the models and other Metrics :-



-Calculating Time for test and train :-



1. **Algorithms tables And charts**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | train |  |  |  | Test |  |  |
| acc | Recall | precision | time | acc | Recall | precision | time |
| **RF** | **1** | **1** | **1** | **53.44803047180176** | **0.9295426452410384** | **0.9295426452410384** | **0.9282027970357631** | **1.9859750270843506** |
| **NB** | **0.7585780525502318** | **0.8192095449133837** | **0.7585780525502318** | **7.526788234710693** | **0.7466007416563659** | **0.7466007416563659** | **0.8173328196055661** | **2.250378131866455** |
| **ANN** | **0.7978361669242658** | **0.7978361669242658** | **0.8338041305818864** | **58.663893699645996** | **0.788627935723115** | **0.788627935723115** | **0.8378443949404994** | **6.808066129684448** |
| **LR** | **1** | **1** | **1** | **24.55480408668518** | **0.8393077873918418** | **0.8393077873918418** | **0.8766767152465844** | **1.792079210281372** |

**Algorithm name :** Z-score normalization + random split

**Algorithm name :** Z-score normalization + Cross validation split

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | acc | Recal | precision | time |
| RF | 0.9141938674579624 | 0.7560240963855421 | 0.8957626868293926 | 276.041241645813 |
| NB | 0.7371414441147379 | 0.8173328196055661 | 0.5204664237108272 | 403.4143795967102 |
| ANN | 0.8921859545004945 | 0.7600401606425703 | 0.8380014133834708 | 275.34473609924316 |
| LR | 0.7116716122650841 | 0.8604417670682731 | 0.4563958570331878 | 388.4143795967102 |

**Algorithm name :** min - max normalization + random split

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | **train** |  |  |  | **Test** |  |  |
| **acc** | **Recall** | **precision** | **time** | **acc** | **Recall** | **precision** | **time** |
| **RF** | **1** | **1** | **1** | **48.44803047180176** | **0.9295426452410384** | **0.9295426452410384** | **0.9282027970357631** | **1.9859750270843506** |
| **NB** | **0.7585780525502318** | **0.8192095449133837** | **0.7585780525502318** | **7.526788234710693** | **0.7466007416563659** | **0.7466007416563659** | **0.8173328196055661** | **2.250378131866455** |
| **ANN** | **0.6124568973152658** | **0.7218963475239857** | **0.7312458976312896** | **63.663893699645996** | **0.788627935723115** | **0.788627935723115** | **0.8378443949404994** | **6.808066129684448** |
| LR | 0.9709428129829984 | 0.9709428129829984 | 0.9708127186477724 | 18.36885356903076 | 0.8603213844252163 | 0.8603213844252163 | 0.8800359877539735 | 1.624168872833252 |

**Algorithm name :** min max normalization + Cross validation split

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | acc | Recal | precision | time |
| RF | 0.9222903245891548 | 0.7502485921735698 | 0.9327626868293926 | 550.2145631645813 |
| NB | 0.8412563365897452 | 0.8125631897456321 | 0.8544664237108272 | 428.2325695967102 |
| ANN | 0.7670149254896372 | 0.0885231496312368 | 0.283614894133834708 | 708.236609924316 |
| LR | 0.8637737925438789 | 0.6804417670682731 | 0.7711268570331878 | 507.412565967102 |

### **A- The best Normalization technique is :-**

**Z-Transform Normalization at it handles outliers , reduces the noise.**

1. **The best classifier as accuracy is**

**Random Forest Classifier . as it is the highest accuracy in 4codes .**

1. **The fastest algorithm :-**

**Naive Baise Algorithm is the fastet one in 3 codes of the 4.**

### **D- Classifiers caused over fitting :-**

**In Min max (hold out ) the LR caused over fitting.**

**In Z score (hold out ) The LR caused over fitting .**