Numpy

.dtype - arraytype - like int, float.

,dtype - change arraytype - like int to float etc.

,base - owns data or not (none is yes)

•shape - returns shape ~~like~~ rowand colomn.

•reshape(6,2) - reshape into dime

reshape(-1) - into 1dim

itrate ndim

for x in np.nditer(arrayname): print(x)

withindexing/pos - np.ndenumerate - in forloop

Joiningarray - np.concatenate

axis = 1 -> col(->)

axis=0 -> row(↓)

np.stack. -> for joining -> 1D-2D

np.hstack -> for horizontal (->) joining

np.vstack -> for vertical (↓) joining

splitting -> np.array_split(myarray, 3) no of array

· dropna( ) - for dropthe ~~array~~ null cell rows

· It return new data set and will not makechanginoriginal dataset.

• dropna(inplace = True) - for drop/remove null cell rows

It not return new data set and makechangesinoriginal dataset

• fillna() - for filling values in data set. - Bydefault fill alldataset

for specified:- colomn

df['colomn name'].fillna(value) - fill specific vale of specific colomn.

Values like

1) mean() - avarage

x=df['colomn name'].mean () → To calulate mean value
df['colomnname'].fillna(x) → Toinserting the calculat vale

2) median() - mid point -

x=df['colomn nane'].median() →
df['colomnname'].fillna(x) -

3) mode - (most recent or repeated vale) or smallest.

x=df['colomn name'].mode()
df['colomn name'].fillna(x) -

df.corr() - for correlations

df.drop(['colomn name'], axis =1) → drop the specific colomn.

x - np.percentile ('var', 75 or any)
which for percentage.

X = random.choice([3,5,7,9], p=[0.1, 0.3, 0.6, 0.0], size(10))
this number are allowed — by this proprablity — size.

random.shuffle(array) - Chang the arrangement of array
it make changes in original array

random.permutation(array) - chang the arrengment of array
it ~~can~~not make change in original array.

x = random.normal(loc=1, scale=2, size(2,3))

loc - mean
scale - (standard Deviation)
size - The shape of return array.

first run on all the models

then we select first three ~~fig~~ highest accurate result.

presition- recomendation system

↳ review negative rating hai and we think it's positive and our money get wasted.

recall- in ~~mad~~ medical case.

↳ testing cor -

~~30~~ ~~40~~ 20 % has covid model said it has corona and they don't take test or medic.

For Linear regression.

1) first we need to divide the indepandend and depandent variable:

x- contain all independend variables

y- contain only dependent variables

lib

- from sklearn.model_selection import train-test-split
- from sklearn.linear_model imorpt LinearRegression

spliting data for training and testing

X-train, x-test, y-train, y-test = train-test-split (x, y, test-size=0.2)

and also use random-state= false)

checkingshape ~~and~~ of train & test data (splited)

1) x-train.shape   3) y-train.shape

2) x-test.shape   4) y-test.shape

linear-regression = LinearRegression() -> creating constructor.

fiting data

linear-regression.fit (x-train, y-train)

testing data ~~f~~or predict. data

(y-pred) predict = linear-regression.predict (x-test)

(x-pred) x predict = linear-regression.predict (x-train)

print by slicing.

print (predict[:5])

This ~~the~~ ~~dredet~~ is predicted value

Y-test -> actual value
Predict -> predicted value

The equation of line.

regression of coffieent

y = mx+c

intercept

- m_value = linear_regression.coef_ (m)
  print(m_value)
- intercept = linear_regression.intercept_ (y)
  print(intercept)

Error = Actual - predicted.

pd.Dataframe({"Actual_value": y_test, "new_prediction": predict,
"Error": y_test - ~~Ac~~ predict})

Train score \ accuracy

train_score = linear_regression.score(x_train, y_train)
print(train_score)

test score \ accuracy

test_score = linear_regression.score(x_test, y_test)

lib

from sklearn.metrics import mean_absolute_error, mean_squared_error
~~accuracy_score~~

- mean_squared_error(y_test, ~~Ac~~ predict~~ed~~)
- mean_absolute_error(y_test, predict)

For Logistic algorithm

lib

```
• from sklearn.model_selection import train_test_split
  from sklearn.linear_model import LogisticRegression
```

Spliting data for training and testing

```
X_train, X_test, Y_train, Y_test = train_test_split(x,y, test_size
                                   =0.2, random_state=False)
```

Constructor

```
logistic_regression = LogisticRegression()
```

Fitting data

```
logistic_regression.fit(x_train, Y_train)
```

predicting values / testing values

```
pred_value = logistic_regression.predict(x_test)
```

lib

```
1) from sklearn.matrics import confusion_matrix, classific-
   cation_report, accuracy_score, mean_squared_error
```

confusion_matrix

```
print(confusion_matrix(y_test, pred_value))
```

classification Report

```
print(classification_report(y_test, pred_value))
```

accuracy- score

(accuracy-score (y-test , predict))

mean square Error

mean_square_error (y-test , pred-values)

Note:- The sum of mean square Error and accuracy-score ~~St~~ must be 100, then our model is correct.

AUC - Roc curve

probability on predicted value.

we have to convert the 2d array to 1d array by [: , 1] as store in new variable

~~y-prid~~ pred_value - prob = logistic-regression.predict-proba (x-test)

lib

from sklearn.metrics import roc-auc-score, roc-curve

:

fpr, tpr , threshold = roc-curve (y-test, ~~p~~ predict-value-prob1)

roc auc score

roc-auc-score (y-test , prid-value-prob1)

for ploting graph

plt.plot (fpr, tpr)

ML model Aws deployment

1) Build the model

2) Export the model using pickle

3) Build a flask website to server the model

4) De.ploy the website on Aws ES2

- Create account on aws
- Create an EC2 instance
- Edit security group
- Download keygen (pem file)
- Download and install putty and winscp
- Upload Flask website to Ec2 using winscp
- install package on ES2 .using putty .

1) python 3) sklearn 4) numpy
2) flask 4) pandas 5) those lab weare used in project.

EC2 - Serive.

1) click on ~~Lainch~~ launch-instance

2) give the name and tag to instance

3) ~~os~~ Select - os image - ubuntu (as required)

~~4~~ instance typ- we select free. (Version)
(optional)

4) create . key-pair.

1) name of key-pair

2) key pairtype - . RSA. - pub - pri - key

3) private key formate - .pem.

Create key - ~~it~~ file is download in folder'

5) copy the file in Project dir. (the mendory file)

6) networksetting - ~~edit~~.- default options
skip-complete.

7) launch - instance

back to ~~Ec2~~ instance . dashbord

#Select the instance. created instance

---

1) winscp'r

1) we need to Hostname that is publicDNS in our created Instance

2) port number - default- (22)

3) username - is ubuntu

4) password - null (need to pass dow key (.ppm))

---

5) 1) click on advance

2) ~~SSHm~~ SSH —> Auth -> brows ~~ke~~ file

4) select the file -> click on ok, ok and Login

and seccion is start -> the two divides Section will open, simply drag and drop the important ~~file~~ and usefull file.

---

file ~~op~~ uploded on server .

artifact	-	pickel file, json.
modal	-	ipynb - file - (optional)
data	-	data file -(optional)
Static	-	Html files., css, js, images, fonts
templats	-	Html files, required.