14 Monday March 2011

Module 3.

March 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
9		1	2	3	4	5	6
10	7	8	9	10	11	12	13
11	14	15	16	17	18	19	20
12	21	22	23	24	25	26	27
13	28	29	30	31			

Week 11 ▪ 73-292

(EDA). Exploratory Data Analysis

Day 14.

Numpy.

- Fast
- Support nd array.
- high level math fn.

Ques. Why do we need Numpy?

If we have 1000x1000 matrix, multiply.

- Python loop > 10min
- Numpy ≈ 0.03s.

Overview.

→ Arrays.

→ Shaping & transposition

→ Mathematical Opesn

→ Indexing & slicing

→ Broadcasting

↓ data of 1 array transfered to another array.

Arrays.

- vectors
- Matrices
- Images
- Tensors
- conv Nets.

} DL part.

Numpy Arrays.

Ex 1. import numpy as np

b = [1, 2, 3, 4, 5]

list ↙

arr = np.array(b)

converted list to np→array

arr

↳ array([1, 2, 3, 4, 5])

Ep.

arr.shape

1D → (5,)

arr.shape[0]

No. of elem in 1D.

#5

arr.shape[1]

no. of col in if array is in 2D form.

arr.shape[2]

no. of channels (if array is in 3D form).

(if it is in 2D, it represent 2D rows.)

2D Array Creation.

b1 = np.array([[1,2,3],[4,5,6],[7,8,9]])

2D. # [[1 2 3]
[4 5 6]
[7 8 9]]

b[1][2] → 6

* array slicing can't be done from backward dirn, it can only be possible from left to right

9
10
11
12
1
3
4
5

numpy functions.

Data analyst may not use it

- np.zeros((4,4)) → 4x4 matrix of 0.
- np.ones((3,6)) → 3x6 matrix of 1
- np.eye(5,5) → dig. 1 & other 0. (identity matrix).

(S) = np.full((3,3),89) → 3x3 matrix with all elem = 89.

np.ones_like(S) → it captures the size of S ie 3x3. You don't need to specify the size in this

creates 3x3 ones matrix.

np.zeros_like(S)

create 3x3 zero matrix.

- a = np.array([[1,2,3,4],[5,6,7,8],[9,10,11,12]]) → 3x4
  b = a[1:3, :2] → [[5,6]
  [9,10]]

[[1,2,3,4]
[5,6,7,8]
[9,10,11,12]]

b[0][0] = 500.

change 0 value to 500 in b & it is impacted on 'a'.

b → [[500, 6, 9, 10]]

a → [[1, 2, 3, 4], [500, 6, 7, 8], [9, 10, 11, 12]]

it is also updated.

b = np.array([1, 1, 2])

[0 1 2]

ss = (a[np.arange(3), b])

[1, 1, 2]

a[0][1]
a[1][1]
a[2,2]

ss → [2, 6, 11]

ss[0] = 200

ss → [200, 6, 11]

Q. Will it make impact on a ?

No impact on a. bcz,

- whenever you make subset with slicing, & then make any change in subset it will reflect in ~~main~~ :a/data
- when you make subset without slicing, it will not make any impact on original data.

place of 2, 6, 11 add +10

a[np.arange(3), b] += 10 ← a=a+10 at the particular place

o/p: 1 12 3 4
500 16 7 8
9 10 21 12

a>2 → gives boolean o/p (T or F)

a[a>2] → gives values with satisfy condn

if a is 2D but this only gives 1D.

Array Math.

default dtype → float.

```
x = np.array([[1,2],[3,4]], dtype=np.float64)
y = np.array([[5,6],[7,8]],  "  )
```

```
np.add(x,y)          # x+y
np.substract(x,y)    # x-y
np.multiply(x,y)     # x*y
np.divide(x,y)       # x/y
np.sqrt(x)
```

```
v = np.array([9,10])
w = np.array([11,12])
```

dot: 9×11 + 10×12

- v.dot(w) # 219
- np.dot(v,w) # 219
- np.dot(x,y) → [[19 22
  43 50]]

```
x = np.array([[1,2,5,9],[3,4,6,7]])
```

37 ← np.sum(x) # compute sum of all elements

[4 6 11 16] ← np.sum(x, axis=0) # compute sums of each coln.

np.sum(x, axis=1) # compute sum of each row.

↓

[17 20]

Transpose.

x → [[1 2
3 4]]

x.T → [[1 3
2 4]]

interchange rows & coln.

#Broadcasting

```
9   np.empty((5,5))  → 5x5 random values.
10  x = np.array([[1,2,3],[4,5,6],[7,8,9],[10,11,12]])
11  y = np.empty_like(x)
12                    empty matrix with same shape as x.
1                     create empty matrix of 3x3.
2   x.shape → (4,3)                ( does not generate 3x3
3   v = np.array([1,0,1])            matrix with random values)
4   for i in range(len(x)):
5=4         y[i,:] = x[i,:] + v[i]
6   print(y)
```

len(x) = 4

= x[0,0]+v[0]
y[0,0] :
= x[0,1]+v[1]
:

[[2 2 4
5 5 7
8 8 10
11 11 13]]

y shape
(4,3)

Day 15.

Combining Arrays.

similar to np.array()

from numpy import array.

from numpy import vstack, hstack → if you want to merge row wise

np.vstack()
np.hstack()

vstack: vertical stack
hstack: horizontal stack

↓ if you want to merge col wise

a1 = array([1,2,3])
a2 = array([4,5,6])
a3 = vstack((a1, a2)) ⟶ [[1,2,3
4,5,6]] merged col wise

a4 = hstack((a1, a2)) ⟶ [1 2 3 4 5 6] merged row wise

o/p is always in 1D.

In vstack,
Both the array should have same no. of ~~dim~~ elements.

It will show an error.
a1 = array([1,2,3]) ⟶ 3 elem.
a2 = array([4,5,6,7]) ⟶ 4 elem
vstack((a1, a2))

for 2D: hstack

a1 = array([[1,2,3],[4,5,6]])
a2 = array([[11,22,33],[44,55,66]])
hstack((a1, a2))

⟶ o/p: 1 2 3 11 22 33
4 5 6 44 55 66

✓ vstack((a1, a2)) executed (no error).

21 Monday
March 2011

March 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
9		1	2	3	4	5	6
10	7	8	9	10	11	12	13
11	14	15	16	17	18	19	20
12	21	22	23	24	25	26	27
13	28	29	30	31			

Week 12 ▪ 80-285

Note.
Conversion is possible only from lower to higher: 1D → 2D ✓
1D → 3D ✗ (not directly)

Array Reshaping.

Reshape 1D to 2D. & 2D to 3D:

```
data = array([11,22,33,44,55])
```

shape. → (5,) 1D data

O/P:

```
data1 = data.reshape(data.shape[0], 1)
```

1 → convert nxt dimension.

shape: (5,1)

2D now

```
[[11
 22
 33
 44
 55]]
```

```
data2 = data.reshape(
        (data.shape[0],
         data.shape[1],1))
```

converted 2D to 3D now.

Pandas.

3 data structures.

→ Series (1D)
→ Dataframe. (2D)
→ Panel. (3D)

Series.

- 1D labelled homogeneous array,
- size immutable.
- values of Data Mutable.

1D array like structure with homogenous data

Dataframe.

- Hetrogeneous data
- size Mutable
- Data mutable.

Pannel.

- Hetrogeneous
- size Mutable
- Data Mutable.

default dtype of all python package is float

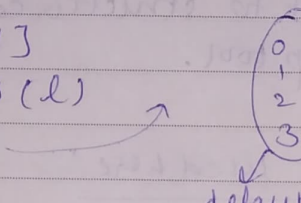
NOTE =

Tuesday
March 2011
22

- Implementation.
  - pd.Series(data, index, dtype, copy)

Ex

```
l = [1,2,3,4]
s = pd.series(l)
s
```



s = pd.Series(data, index = [9,10,11,12])

9 1
10 2
11 3
12 4
} customized o/p.

s[0] → error
s[11] → 3

data = {'a': 0, 'b': 1, 'c': 2}

s = pd.series(data)

a 0
b 1
c 2

In dictionary, you can access element by default indexing or user defined.

s['a'] = s[0] → 0
s['b'] = s[1] → 1

s[::-1] →
c 2
b 1
a 0

Day-16.

Dataframe.

- explore or handle 2d data.
- both size & info(data) are mutable.

c:/users/... diabetes-with-name.csv'

df = pd.read_csv('c:\\users\...diabetes.csv')

don't use backward slash (use '/' forward slash.)

Pandas don't support Backward('\') slash.

23 Wednesday March 2011

pandas documentation for reference.

March 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
9		1	2	3	4	5	6
10	7	8	9	10	11	12	13
11	14	15	16	17	18	19	20
12	21	22	23	24	25	26	27
13	28	29	30	31			

Week 12 ▪ 82-283

- ML algo. supports only structured data.
  Unstructured data can't be used directly,
  convert it to structured by hadoop tool.
  Ex. wordfile →

- Datafile or Dataset used here:
  - Diabetes with names.csv.

- Null values affect the average. (Statistics gets differ.)
  Ex. You have 15 rows & 2 are Null. (NaN).
  (taking null values) Avg. by comp. for 15 rows = 34.25 → sum/15
  Actual avg = sum of 13 rows / 13
  removing null values, then cal. avg.

df.isna().sum()
df.isnull().sum()
→ calculate missing values in a row & prints sum of each row seperately.
Both are same.

df.isna().sum().sum() → calculate total no. of missing values in entire dataset.

df.fillna() → this fn fills every NaN value with that value.
Ex. df.fillna(df.mean())

df.info() → info. of dataframe. (get all file info.)
(gives datatypes, total values in a particular row..)

len(df) → gives total no. of rows in dataset.

df.size → row x col

Ex 8 rows, 5 col : o/p = 40 (8x5)

df.shape → gives shape of dataset.

Ex (8,5)

df.describe() → returns statistical summary of dataset.

Ques If NaN value are not there in dataset, instead of NaN any other variable like '?', 'unknown' etc is there will isnull().sum() gives same result?

Soln NO,

find '?' in dataset by method, replace it with NaN & then try to fill it.

Alternative,

while loading dataset, define there which type of null val you have.

Ex df = pd.read_csv('___.csv', na_values='?')

we have ? as Null value in dataset.

df.isnull().sum()

↳ You will get the count of null values.

Soln 1 file → Dataset (titles.csv) → movie title dataset.

df1 = pd.read_csv('titles.csv', index_col=None)

df1.head()

↳ top 5 rows.

```
df3 = pd.read_csv('cast.csv', index_col=None)
```

Q1 earliest 2 film in title dataset

```
df2.sort_values('year').head(2)
```

always sort in ascending order.
(goes to higher).

use sen.

Q2. How many movies have title 'Hamlet'?

```
len(df2[df2.title == 'Hamlet'])
```

Q3. When was first movies titled 'Hamlet' made?

```
df2[df2.title == 'Hamlet'].sort_values('year').head(1)
```

(Check Notes given)

Day 17.

Exploratory Data analysis.

Dataset → Diabetes with names.csv

```
df = pd.read_csv('_____.csv')
df.head()

df.isna().sum()
```

Handling Missing Values.

- should have domain knowledge.
- when we encounter huge no. of nan values, we use to fill nan by calling fillna().
- less no. of nan, if you wish you can drop.

create droped record as subset.

Saturday 26

March 2011

Week 12 ▪ 85-280

how values should be deleted.

`df.dropna(self, axis=0, how='any', thresh=None, subset=None, inplace=True)`

axis=0 → drop row nan

'any' or 'all'

thresh → you can set the limit to del missing value.

NaT → Not a timestap.
(It is used where time/date is used).

`df.dropna()` → It drops all rows where atleast1 element is missing.

`df.dropna(how='all')` → It will drop when in a ~~col~~/row all the values are NaN.

`df.dropna(thresh=2)` → keep rows with atleast 2 non-Na values.

- Filling missing values.

`df.fillna(df.mean())`

`df.fillna(method='pad').head()`

(Previous value is placed in place of NaN) takes before value & place over it

Sunday 27

86-279

padding

forward (ffill) (default)

backward (bfill)

syntax: `fillna(method='bfill')`

Ex =	6	pad 6	bfill 6
	NaN	6	8
	8	8	8

28 Monday

March 2011

Week 13 ▪ 87-278

bfill will Notfill.

NaN → 1st

NaN → Not fill in case of bfill

Ex = 6 8 2

March 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
9		1	2	3	4	5	6
10	7	8	9	10	11	12	13
11	14	15	16	17	18	19	20
12	21	22	23	24	25	26	27
13	28	29	30	31			

To handle value of first & last value (exception).

You can use both ffill & bfill (but not at a time, use simultaneously.)

dynamic way of filling values.

df.interpolate() → it will fill missing value itself.

Problem: We can't judge which method is used to fill values.

→ We should not give entire data to M/C. bcz if model is not have nice accuracy. You will not be able to check why it is not giving accuracy.

} If you handle yourself you can increase or decrease the accuracy.

interpolate feature.

- spline → use circle eqn
- polynomial → Magnitude value in logic..
- akima
-
-

EDA → Know Nature of Data.

Plot Graph → to get insights.

data.describe().T → Transposed Matrix.

data.hist(figsize=(12,10))

↓

density of record.

Day-18

data.boxplot(figsize=(12,6))

Try to explore IQR (25%, 50%, 75%)

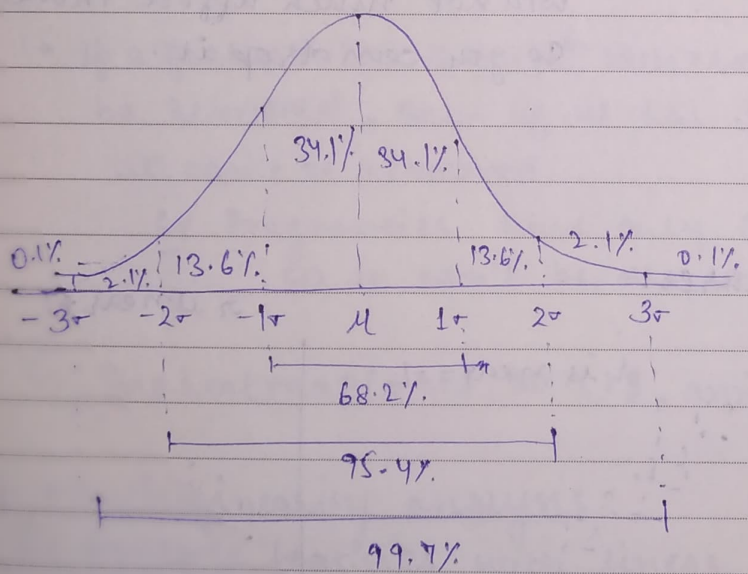
Used for outlier Detection / noise
- ~~used to know noise~~ check five point summary.

Check one by one:

```
sns.boxplot(df['Glucose'])
sns.boxplot(df['Pregnencies'])
⋮
```

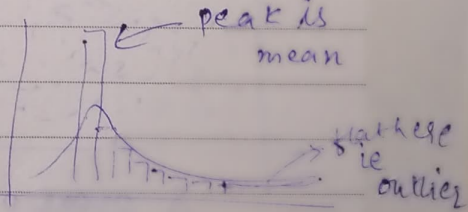
Inference from Boxplot.

Max outlier, Min outlier, Moderate outlier, No outlier



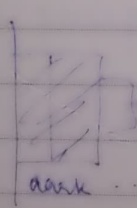
Distribution is used to give dataflow of feature

sns.distplot(df['col'])
↳ gives distribution.



sns.boxenplot(df['col'])

color variance outliers →



data distribution can be identified by color of graph

4 has outliers
2 has moderately linearity.

30 Wednesday March 2011

Week 13 • 89-276

Q. What is Max. outliers?
value above $\mu \pm 2\sigma$.

Q. Should we remove or retain/as it is?

March 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
9		1	2	3	4	5	6
10	7	8	9	10	11	12	13
11	14	15	16	17	18	19	20
12	21	22	23	24	25	26	27
13	28	29	30	31			

When you have max. outliers. → keep the moderate outliers which are near to max. pt.

→ drop extreme outlier.?

- No, if we have not so large data.

  Ex 10,000 insulin patients, 100 people have 800 as insulin level. (So it is advised to keep outlier)

- Yes, if we have very large amount of data.

  Ex remove insulin patient, 1-2 people have 800 level of insulin. It will not much affect model, so you can drop it.

outlier should be retained bcz they restricts the (limit) performance of model.
→ overfitting se bchata hai model ko.

(You should have Domain knowledge for outlier treatment)

max. outlier.

(Rare) Too extreme - Drop it

moderate outliers

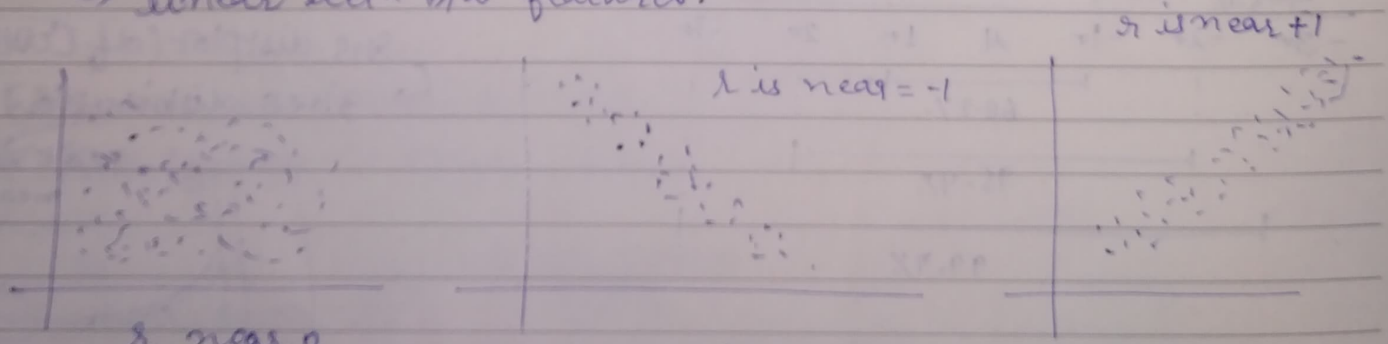
Retain it

Minimum outlier

keep it / retain it

Correlation.

↳ linear reln b/w features.



coeficient of relⁿ (Pearson's coefficient):

$$r = \rho(x,y) = \frac{cov(x,y)}{std.x \cdot std.y}$$

-1 → strong -ve corr.
+1 → strong +ve corr.
0 → no correln.

Tips for taking decision on corr. coeff.

-0.2 to +0.2 → consider as 0 (no corr.)
(-0.3 - -0.6) to (+0.3 - +0.6) → moderate reln (either +ve or -ve)
(-0.6 - -0.8) to (0.6 ~ +0.8) → good corr.
> 0.8 in both sides → strong corr.

Select imp. features

what decision to take →
→ no corrln : drop these features → (Feature selection technique)

- If a feature has 0 corrln with every other feature then it can be removed, But if it has corrln with atleast 1 feature it can't be removed.

Ex Pregnencies have only corrln with age (all other are ≈0)
so it can't be droped.

```
sns.heatmap(data.corr(), annot = True)
```

Multicolinearity analysis.
↳ multiple features with linear reln.

sns.pairplot(data) → You can't determine reln.
or
* sns.pairplot(data, hue = 'outcome')

plt.xlabel('_____')
plt.ylabel('_____')

April 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
13					1	2	3
14	4	5	6	7	8	9	10
15	11	12	13	14	15	16	17
16	18	19	20	21	22	23	24
17	25	26	27	28	29	30	

```
sns.barplot(x=df['coln1'], y=df['coln2'])

sns.scatterplot(x=df['coln1'], y=df['coln2'])

sns.pointplot(x=df['coln1'], y=df['coln2'])
```

gives trend.

```
df.coln.value_counts()
or
df['coln'].value_counts()
```

} to counts no. of values which belongs to some category.

```
pd.crosstab(df.coln1, df.coln2)
```

acts like pivot table.

counts value of each coln.

ex

sal → coln1 , department → coln2

Dep. / Sal.	IT	HR	CSE	EC	ME	CIVIL
High	83	51	225	80	74	68
low	609	368	364	180	402	451
med.	535	180	370	225	376	363

Scatter plot with multivariate analysis ,

```
sns.lmplot(x='coln1', y='coln2', data=df, hue='col',
           fitreg=False)
plt.show()
```

it is like pairplot with 'hue'.

- dropping coln.

```
x = df.drop('coln', axis=1)
```

Concepts of ML & Linear Regression.

Day-19.

Supervised — Regression, classification

Unsupervised → clustering.

Predictive Modelling

- Regression → x (Numerical/categorical) → y (numerical)
  - Ex: Stock Market Predn, House Predn, Salary Predn
  - Target (y).
- Classification → x (numerical/categorical) → y (categorical)
  - Ex: Predict something as High, Low, Person, diabetes or not

Linear Regression.

Lin. if something ↑ or ↓ it is linear.

predicting real no.

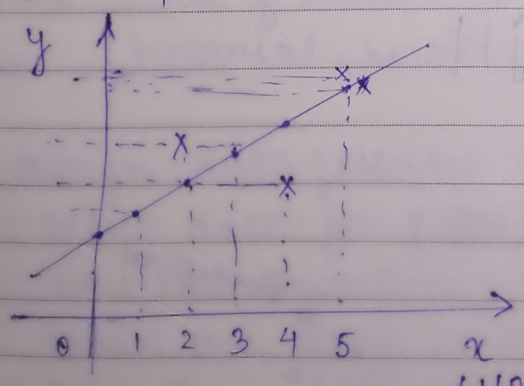
$y = mx + c$ (line eqn). — m: slope, c: intercept.

$$m = \frac{n\Sigma xy - \Sigma x \Sigma y}{n\Sigma x^2 - (\Sigma x)^2}$$

n: no. of records

$$c = \frac{1}{n}(\Sigma y - m\Sigma x)$$

(O/P) → depn var.



Σx	x	y	xy ↓	x^2 ↓
	0	2		
	1	3		
	2	5		
	3	4		
	4	6		
Σ	—	—	—	—

Put all the values in m & c, & find ŷ.

Sunday 3
93-272

error → predicted - actual.

Sol. After calculation,

$\hat{y} = 0.9x + 2.2$

x	y	$\hat{y}$	$y - \hat{y}$
0	2	2.2	$2 - 2.2 = -0.2$
1	3	3.1	
2	5	4	
3	4	4.9	
4	6	5.8	

we can't measure error in -ve.

Actual, predicted

Estimation.

Linear Regression Performance metrics / cost Fn / Error ∧

Error = $y - \hat{y}$

Total error = $\Sigma(y - \hat{y})$

Avg error / Mean error = $\frac{1}{n} * \Sigma(y - \hat{y})$

Mean squared error (MSE) = $\frac{1}{n} * (\Sigma(y - \hat{y}) * (y - \hat{y})$

Mean Absolute error = $\frac{1}{n} * |\Sigma(y - \hat{y})|$

modulus.

Root Mean SQuared error (RMSE) = sqrt (MSE)

Imp: R-Square Value : $R^2 = 1 - \frac{\Sigma(y_i - \hat{y}_i)^2}{\Sigma(y_i - \bar{y})^2}$

Predicted.

mean value

b/w lies 0 to 1

May 2011

Wk	Mo	Tu	We	Th	Fr	Sa	Su
17/22	30	31					1
18	2	3	4	5	6	7	8
19	9	10	11	12	13	14	15
20	16	17	18	19	20	21	22
21	23	24	25	26	27	28	29

Tuesday

April 2011

Week 14 ▪ 95-270

Implementation steps.

1. Data gathering for problems
   ↳ collect the data

2. Requirement Analysis to be done.
   ↳ what we want to explore.
   (what is objective).

3. Data Exploration using pandas perform EDA
   ↳ Data preprocessing / Wrangling...
   cleaning data (find missing values...)
   try to plot datapoints ⟶ came to know data patterns.
   (We can't plot all data features, If 2 features → plot.
   n features → you can't plot.)

4. Data preprocessing - check missing values.
   - NAN & NAT value handling.
     ⟶ isna(), notna()
     ⟶ fillna() → use various methods to fill NaN
     ⟶ interpolation, bfill, ffill, pad..

5. Find nature of data distribution by various methods such as binomial, bernoli, uniform ... soon
   (* ↙ = distplot)

6. assign data features to X-input, y output
7. split data for training & testing
8. call the algorithm from SKlearn
9. train the data using algo. using fit method.
10. test the model using test data
11. call the performance metrics Ex R^2, RMSE, accuracyscore etc...

Adjusted R^2

when we are not satisfied with R^2 we go for adjusted R^2 (very precised measures).

$$= 1 - \frac{\frac{\Sigma(y_i - \hat{y}_i)^2}{n-p-1}}{\frac{\Sigma(y_i - \bar{y})^2}{n-1}}$$

6 Wednesday April 2011

Simple Linear Regression

Implementation.

Dataset used: Auto insurance payment.

```
df = pd.read_csv('auto-insurance-payment.csv')
df.head()

df.dtypes
df.info()
df.describe()

EDA.
    df.hist()
    df.corr()
    sns.scatterplot(df['claims'], df['Payment'])
    ↳ check linearity.
    sns.pairplot(df)
    sns.boxplot() → check do we need to treat outlies?

x = df.iloc[:,:1]
y = df.iloc[:,1:]
```

Split Records

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(x, y,
                    test_size = 0.2 , random_state = 42)
```

20% to test size
80% to train set

data.shape
x_train.shape
x_test.shape

Model Building.

```
from sklearn.linear_model import LinearRegression()
lin = LinearRegression()
```

Train Model.

```
lin.fit(x_train, y_train)
~~lin.coeff~~ lin.coef_
~~lin.ite~~ lin.intercept_
```

Visualize trainingset result.

```
plt.scatter(x_train, y_train, color='red')
plt.plot(x_train, lin.predict(x_train), color='green')
plt.show()
```

Test Model.

```
ypred = lin.predict(x_test)
```

Visualize test set result

```
plt.scatter(~~x_train, y_train~~ x_test, y_test, color='blue')
plt.plot(x_test, lin.predict(x_test), color='green')
plt.show()
```

include →
plt.title('')
plt.xlabel('')
plt.ylabel('')

Estimate cost

from sklearn.metrics import mean_squared_error, r2_score.

RMSE = np.sqrt(mean_squared_error(ytest, ypred))

Imp. ★ R2 = r2_score(y_test, y_pred)

Ques. How to predict for unseen value ?

unseen_pred = lin.predict(np.array([[108]]))

↳ give any value it will predict o/p.

Multiple Regression (Bivariate Analysis). Day-20.

Linear.

Dataset → 'health insurance cost'

19 coln only.

df=pd.read_csv('health_insurance_mising_data.csv')

df.head()

df.dtypes. → Algo. don't take object values. so convert 'obj' to 'numeric'.

df.info()

df.describe()

df.isna().sum()

drop irrelevant coln. (check ~~which~~ by stats. or domain knowledge or by plotting graphs..) which coln is irrelevant for predicting target.

df = ~~data~~ df.drop(['coln','coln'], axis=1)

df = df.fillna(method = 'bfill') → anything you can fill with mean... etc

Q. When to do Normalization?

if instances of features differs numerically as larges value. Ex one col^n has value in single digit, other has 5, 6, digits.. Statistically higher digits features dominates lower dig. values ie,
which feature has higher value thats std, mean etc can dominate all other features.

By Normalization → Performance of model ↑es.

Standard Scalar. →

$$Z = \frac{x - \text{mean}}{\text{std}} \qquad \frac{x - \mu}{\sigma}$$

```
from sklearn.preprocessing import StandardScaler.
sc = StandardScaler()
df1 = sc.fit_transform(df)
```

It transforms our output also.

Model1.

```
x = df.iloc[:, 0:3]
y = df.iloc[:, -1:]
```

} Here sex, smoker, region col^n are droped. & no standard scaling is applied.

Split it in 75-25%

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y,
                                   test_size = 0.25, random_state
                                   = 42)
```

Train Model.

```
from sklearn.linear_model import LinearRegression
multiple = LinearRegression()
```

we have parameter
normalize = False default.
(we don't need to seperately do normalization, when we make it True, algo will automatically do normalization)

```
multiple.fit(x_train, y_train)
```

Test model.

```
y_pred = multiple.predict(x_test)
```

on unseen data.

```
unseen = multiple.predict(np.array([[19, 33, 0]]))
unseen
```

↳ -134.345

model is performing too bad.
(-ve prediction).

Evaluation.

```
from sklearn.metrics import r2_score
r2_score(y_test, y_pred)
```

-2.1527
(worst model.)

here R^2 is negative i.e. it is too bad model.

$0 < R^2 < 1$

df
~~data~~.corr()
↳ no corr(n b/w features.
∴, we do EDA process in starting to know whether our data is compatible or we can go further with this or not.

df~~data~~.hist(figsize = (10,8))
~~data~~ sns.pairplot(df)

df2 = pd.read_csv('health-insurance.cost.csv')
df2.shape ↳ (1338,7)

converting categorical to Numerical →

```
from sklearn.preprocessing import LabelEncoder
lab = LabelEncoder()
df2['smoker'] = lab.fit_transform(df2['smoker'])
```

converting int/float ... into category type:

- df['coln'] = df['coln'].astype('category')
- or ≫ df['coln'] = ~~df~~ pd.Categorical(df['coln'])

category types can't be in statistical summary.

Exploring categorical feature.

```
df['coln'].value_counts(normalize = False)
sns.countplot(x = 'Gender', data = df)
```

any coln name.

Boxplot with 2 features -

```
sns.boxplot(x = df['col1'], y = df['col2'])
```

* Multivariate Analysis.

Ex

```
g = sns.FacetGrid(df, col='Married', row='Gender')
g.map(sns.boxplot, 'Losses')
```

pivot table → summarization of more than 2 or 3 feature

Ex

```
pd.pivot_table(data=df, values='Losses', index='
    No.of vehicle', columns='Gender',
    aggfunc='mean')
```

convert categorical → dummy var.

```
x = pd.get_dummies(x, drop_first=True)
```

* OLS Method

OLS → ordinary least square

```
import statsmodels.api as sm
lm = sm.OLS(y_train, x_train)
lm2 = lm.fit()
lm2.summary()        → check r² value.
```

• Linear Regression.

```
lm3 = LinearRegression()
lm3.fit(x_train, y_train)
y_pred = lm3.predict(x_test).   check r² then.
```