Gradient Boosting

Concept Explained

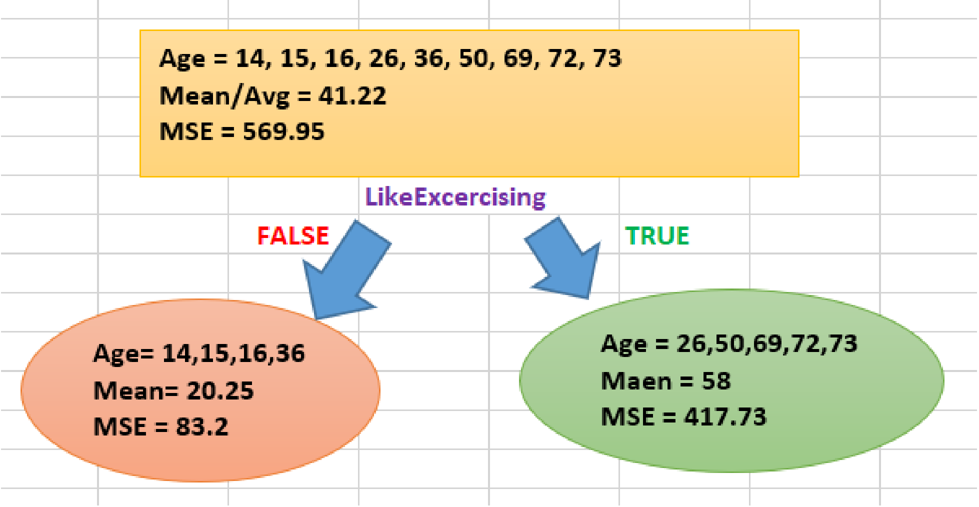
Gradient Boosting- Regression

Error ^2

B	C	D	E	F	G	H	I
						0_level model	
	LikesExcercising	GotoGym	DrivesCar	Age		pred0	resd0
1	FALSE	TRUE	TRUE	14	741.0494	41.22	-27.22
2	FALSE	TRUE	FALSE	15	687.6049	41.22	-26.22
3	FALSE	TRUE	FALSE	16	636.1605	41.22	-25.22
4	TRUE	TRUE	TRUE	26	231.716	41.22	-15.22
5	FALSE	TRUE	TRUE	36	27.2716	41.22	-5.22
6	TRUE	FALSE	FALSE	50	77.04938	41.22	8.78
7	TRUE	TRUE	TRUE	69	771.6049	41.22	27.78
8	TRUE	FALSE	FALSE	72	947.2716	41.22	30.78
9	TRUE	FALSE	TRUE	73	1009.827	41.22	31.78
			total=	371	5129.556		
			avg=	41.22222	**569.9506**		
					=MSE		

1st Estimator

These will be used for 2nd estimator



	J	K	L	M	N	O	P	Q	R
1		1ST ESTIMATOR				pred1			
2		LikesExcercising	GotoGym	DrivesCar	Age	AVG		MSE	resd1
3	1	FALSE	TRUE	TRUE	14	20.25	39.0625		-6.25
4	2	FALSE	TRUE	FALSE	15		27.5625		-5.25
5	3	FALSE	TRUE	FALSE	16		18.0625		-4.25
6	5	FALSE	TRUE	TRUE	36		248.0625		15.75
7						total =	332.75	83.1875	
8									
9	4	TRUE	TRUE	TRUE	26	58	1024		-32
10	6	TRUE	FALSE	FALSE	50		2500		-8
11	7	TRUE	TRUE	TRUE	69		4761		11
12	8	TRUE	FALSE	FALSE	72		5184		14
13	9	TRUE	FALSE	TRUE	73		5329		15
14						total =	2088.667	417.7333	
15								500.9208	

1st estimator

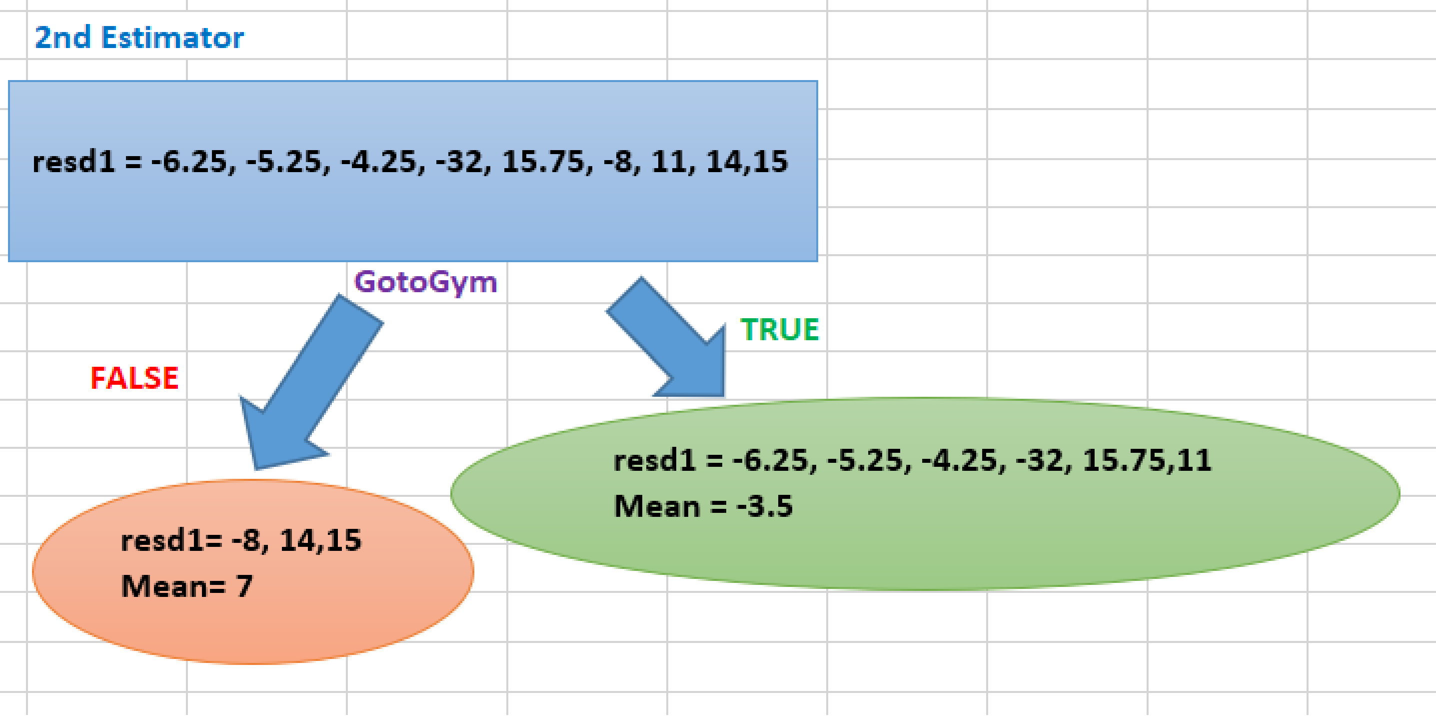
	J	K	L	M	N	O	P	Q	R
1		1ST ESTIMATOR				pred1			
2		LikesExcercising	GotoGym	DrivesCar	Age	AVG		MSE	resd1
3	1	FALSE	TRUE	TRUE	14	20.25	39.0625		-6.25
4	2	FALSE	TRUE	FALSE	15		27.5625		-5.25
5	3	FALSE	TRUE	FALSE	16		18.0625		-4.25
6	5	FALSE	TRUE	TRUE	36		248.0625		15.75
7						total =	332.75	83.1875	
8									
9	4	TRUE	TRUE	TRUE	26	58	1024		-32
10	6	TRUE	FALSE	FALSE	50		2500		-8
11	7	TRUE	TRUE	TRUE	69		4761		11
12	8	TRUE	FALSE	FALSE	72		5184		14
13	9	TRUE	FALSE	TRUE	73		5329		15
14						total =	2088.667	417.7333	
15								500.9208	

2^{nd} Estimator : GotoGym

	resd1
1	-6.25
2	-5.25
3	-4.25
5	15.75
4	-32
6	-8
7	11
8	14
9	15

2ND ESTIMATOR

	LikesExcercising	GotoGym
6	TRUE	FALSE
8	TRUE	FALSE
9	TRUE	FALSE
1	FALSE	TRUE
2	FALSE	TRUE
3	FALSE	TRUE
4	TRUE	TRUE
5	FALSE	TRUE
7	TRUE	TRUE



2nd Estimator

	J	K	L	M	N	O	P	Q	R	S
16		2ND ESTIMATOR								
17		LikesExcercising	GotoGym	DrivesCar	Age	pred1	pred2	FinalPred	FinalResd	e^2
18	6	TRUE	FALSE	FALSE	50	58	7	65	-15	225
19	8	TRUE	FALSE	FALSE	72	58	7	65	7	49
20	9	TRUE	FALSE	TRUE	73	58	7	65	8	64
21	1	FALSE	TRUE	TRUE	14	20.25	-3.5	16.75	-2.75	7.5625
22	2	FALSE	TRUE	FALSE	15	20.25	-3.5	16.75	-1.75	3.0625
23	3	FALSE	TRUE	FALSE	16	20.25	-3.5	16.75	-0.75	0.5625
24	4	TRUE	TRUE	TRUE	26	58	-3.5	54.5	-28.5	812.25
25	5	FALSE	TRUE	TRUE	36	20.25	-3.5	16.75	19.25	370.5625
26	7	TRUE	TRUE	TRUE	69	58	-3.5	54.5	14.5	210.25
27										1742.25
28									MSE =	193.5833
29										

Look at the reduction in mse!

Libraries

```
# Jesus is my Saviour!

import numpy as np
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import GridSearchCV

# Let us create the Data-Frame for above
X=pd.DataFrame({'LikesExercising':[False,False,False,True,False,True,True,True,True],
                'GotoGym':[True,True,True,True,True,False,True,False,False],
                 'DrivesCar':[True,False,False,True,True,False,True,False,True]})
```

```
In [3]: X.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   LikesExercising  9 non-null      bool
 1   GotoGym          9 non-null      bool
 2   DrivesCar        9 non-null      bool
dtypes: bool(3)
memory usage: 155.0 bytes
```

Our Target Variable

```
In [4]: Y=pd.Series(name='Age',data=[14,15,16,26,36,50,69,72,73])

In [5]: Y
Out[5]:
0     14
1     15
2     16
3     26
4     36
5     50
6     69
7     72
8     73
Name: Age, dtype: int64
```

```
In [3]: X.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9 entries, 0 to 8
Data columns (total 3 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   LikesExercising  9 non-null     bool
 1   GotoGym          9 non-null     bool
 2   DrivesCar        9 non-null     bool
dtypes: bool(3)
memory usage: 155.0 bytes
```

Lets make 0 and 1 for categorical variables

```
In [6]: LE=LabelEncoder()

In [7]: X['LikesExercising']=LE.fit_transform(X['LikesExercising'])

In [8]: X['GotoGym']=LE.fit_transform(X['GotoGym'])

In [9]: X['DrivesCar']=LE.fit_transform(X['DrivesCar'])

In [10]: X
Out[10]:
   LikesExercising  GotoGym  DrivesCar
0                0        1          1
1                0        1          0
2                0        1          0
3                1        1          1
4                0        1          1
5                1        0          0
6                1        1          1
7                1        0          0
8                1        0          1
```

After

Before

```
# Let us create the Data-Frame for above
X=pd.DataFrame({'LikesExercising':[False,False,False,True,False,True,True,True,True],
                'GotoGym':[True,True,True,True,True,False,True,False,False],
                 'DrivesCar':[True,False,False,True,True,False,True,False,True]})
```

GB Model with 2 estimators

```
#Lets build 2 estimators

# 1) Let us now use  GradientBoostingRegressor with 2 estimators to
#train the model and to predict the age for the same inputs.
GB=GradientBoostingRegressor(n_estimators=2)
GB.fit(X,Y)
Y_predict=GB.predict(X) #ages predicted by model with 2 estimators
Y_predict
'''
array([38.14 , 36.335, 36.335, 42.415, 38.14 , 44.98 , 42.415, 44.98 ,
       47.26 ])'''

# MSE of residuals
MSE_2=(sum((Y-Y_predict)**2))/len(Y)
print('MSE for two estimators :',MSE_2)
#Output: 427.78
```


```
# MSE of residuals
MSE_2=(sum((Y-Y_predict)**2))/len(Y)
print('MSE for two estimators :',MSE_2)
#Output: 427.78
```

GB Model with 3 estimators

```
#Lets build 3 estimators

GB3=GradientBoostingRegressor(n_estimators=3)
GB3.fit(X,Y)
Y_predict3=GB3.predict(X) #ages predicted by model with 3 estimators
Y_predict3
'''
array([36.826 , 34.2515, 34.2515, 42.9235, 36.826 , 46.582 , 42.9235,
       46.582 , 49.834 ])
'''
MSE_3=(sum((Y-Y_predict)**2))/len(Y)
print('MSE for three estimators :',MSE_3)
#Output: 376.25
```

Observe the reduction

```
# MSE of residuals
MSE_2=(sum((Y-Y_predict)**2))/len(Y)
print('MSE for two estimators :',MSE_2)
#Output: 427.78
```

With 50 estimators

```
# 3) GB Model with 50 estimators

GB50=GradientBoostingRegressor(n_estimators=50)
GB50.fit(X,Y)
Y_predict50=GB.predict(X) #ages predicted by model with 50 estimators
Y_predict50

MSE_50=(sum((Y-Y_predict50)**2))/len(Y)
print('MSE for fifty estimators :',MSE_50)
'''
MSE for fifty estimators : 427.78405555555554
'''
```

More numbers of estimators is not a guarantee of better mse!

Grid Search

```
from sklearn.model_selection import GridSearchCV
model=GradientBoostingRegressor()
params={'n_estimators':range(1,200)}
grid=GridSearchCV(estimator=model,cv=2,param_grid=params,scoring='neg_mean_squared_error')
grid.fit(X,Y)
print("The best estimator returned by GridSearch CV is:",grid.best_estimator_)
'''
The best estimator returned by GridSearch CV is:
    GradientBoostingRegressor(n_estimators=8)'''
```

Best Model

```
GB=grid.best_estimator_
GB.fit(X,Y)
Y_predict=GB.predict(X)
Y_predict

MSE_best=(sum((Y-Y_predict)**2))/len(Y)
print('MSE for best estimators :',MSE_best)
# 233.15

# learners: your results may vary!
```

What was 4th estimator?

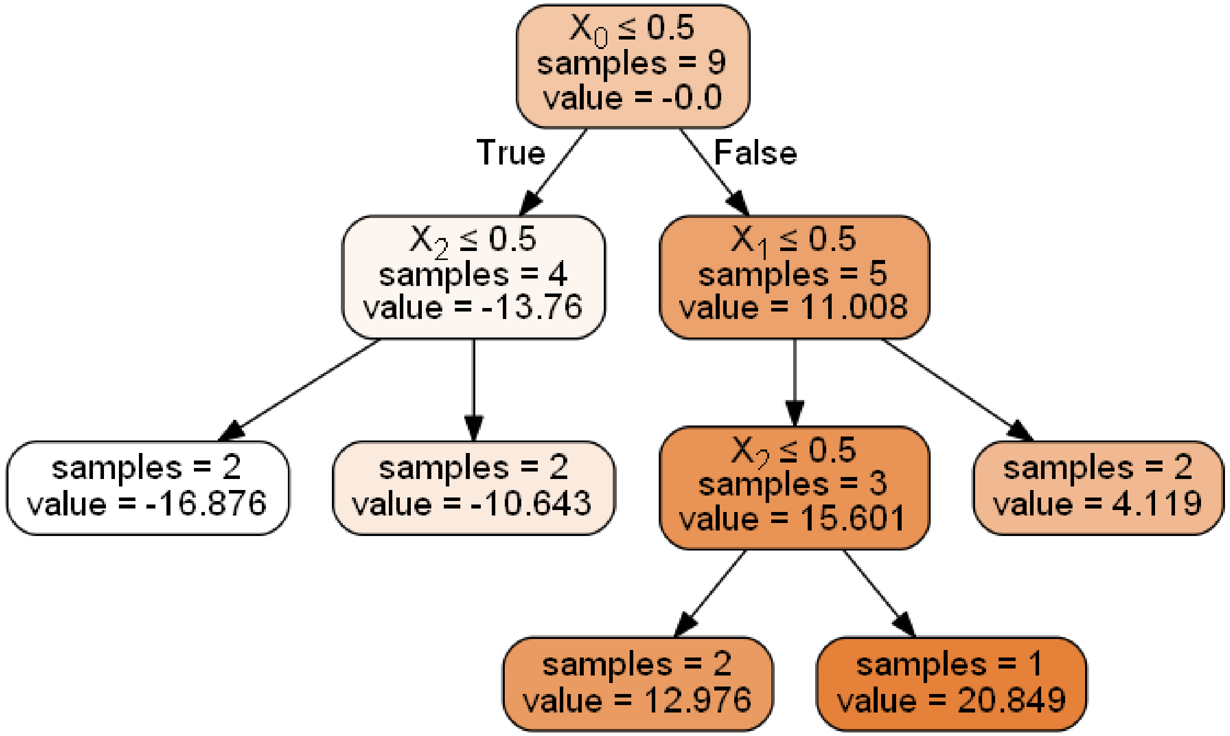
```
In [65]: sub_tree_4 = GB.estimators_[4, 0] # GB is the best model

In [66]: from pydotplus import graph_from_dot_data

In [67]: from IPython.display import Image

In [68]: dot_data = export_graphviz(
    ...:     sub_tree_4,
    ...:     out_file=None, filled=True, rounded=True,
    ...:     special_characters=True,
    ...:     proportion=False, impurity=False, # enable them if you want
    ...: )
    ...: graph = graph_from_dot_data(dot_data)
    ...:
    ...: Image(graph.create_png())
```

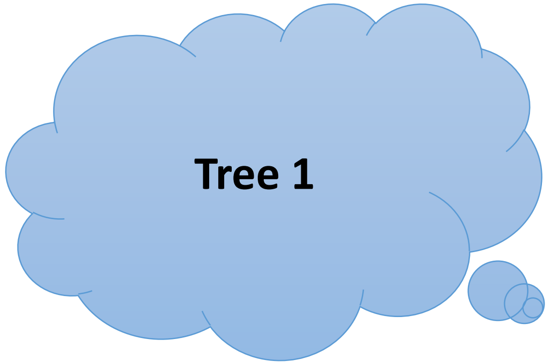
4th estimator

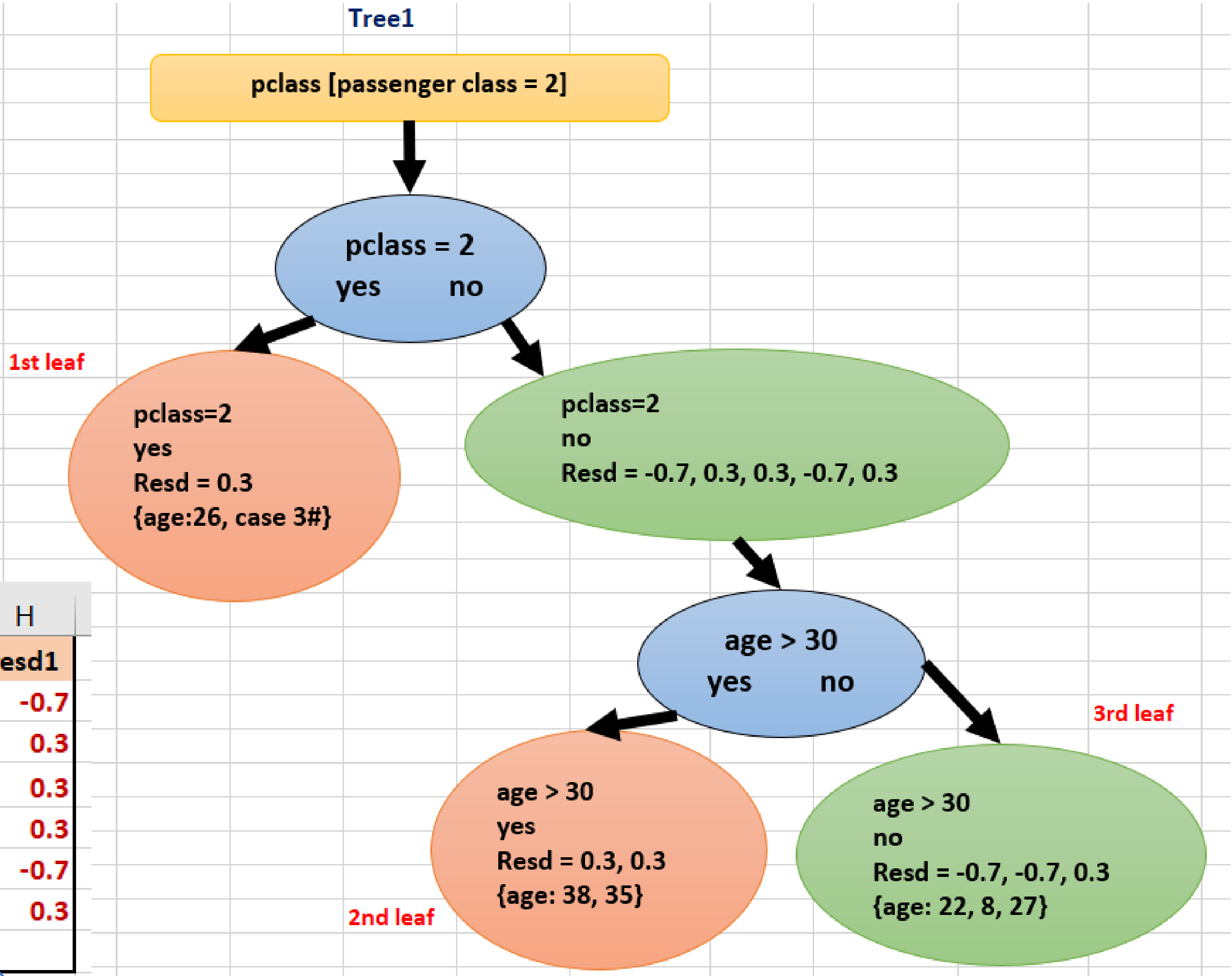


Classification

Data

	A	B	C	D	E	F
1		pclass	age	fare	sex	survived
2	1	3	22	7.25	m	0
3	2	1	38	71.28	f	1
4	3	2	26	7.93	f	1
5	4	1	35	53.1	f	1
6	5	3	8	21.07	m	0
7	6	3	27	11.13	f	1
8						1= survived, 0= not



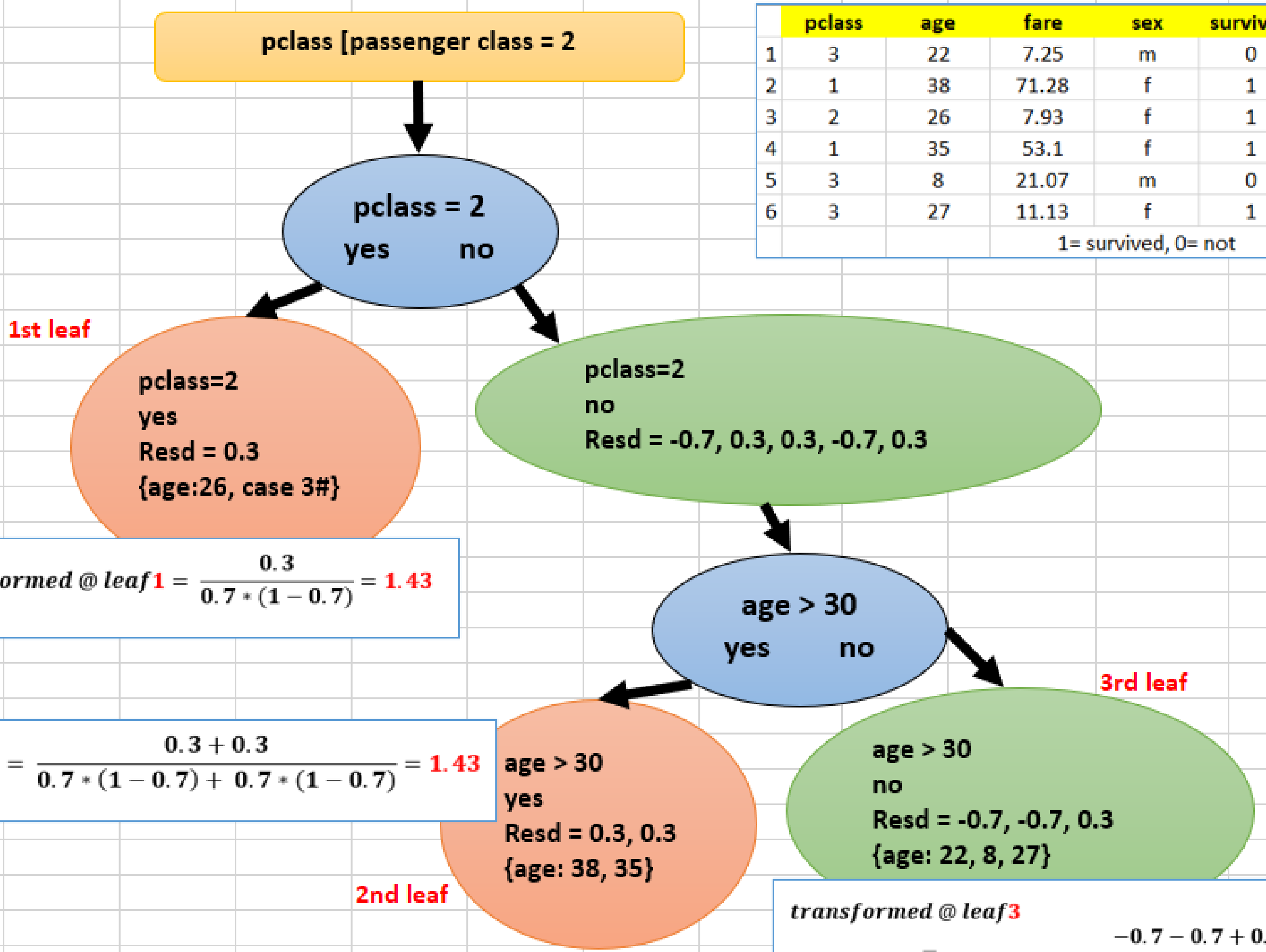


Base Model/Null Model/Zero Model/Reference Model

	A	B	C	D	E	F	G	H
1		pclass	age	fare	sex	survived	pred1 (prob)	resd1
2	1	3	22	7.25	m	0	0.7	-0.7
3	2	1	38	71.28	f	1	0.7	0.3
4	3	2	26	7.93	f	1	0.7	0.3
5	4	1	35	53.1	f	1	0.7	0.3
6	5	3	8	21.07	m	0	0.7	-0.7
7	6	3	27	11.13	f	1	0.7	0.3
8						1= survived, 0= not		

$$log(odds) = Log\left(\frac{4}{2}\right) = LN(2) = 0.693147 = 0.7$$

Transformed Tree



	pclass	age	fare	sex	survived	pred1 (prob)	resd1
1	3	22	7.25	m	0	0.7	-0.7
2	1	38	71.28	f	1	0.7	0.3
3	2	26	7.93	f	1	0.7	0.3
4	1	35	53.1	f	1	0.7	0.3
5	3	8	21.07	m	0	0.7	-0.7
6	3	27	11.13	f	1	0.7	0.3

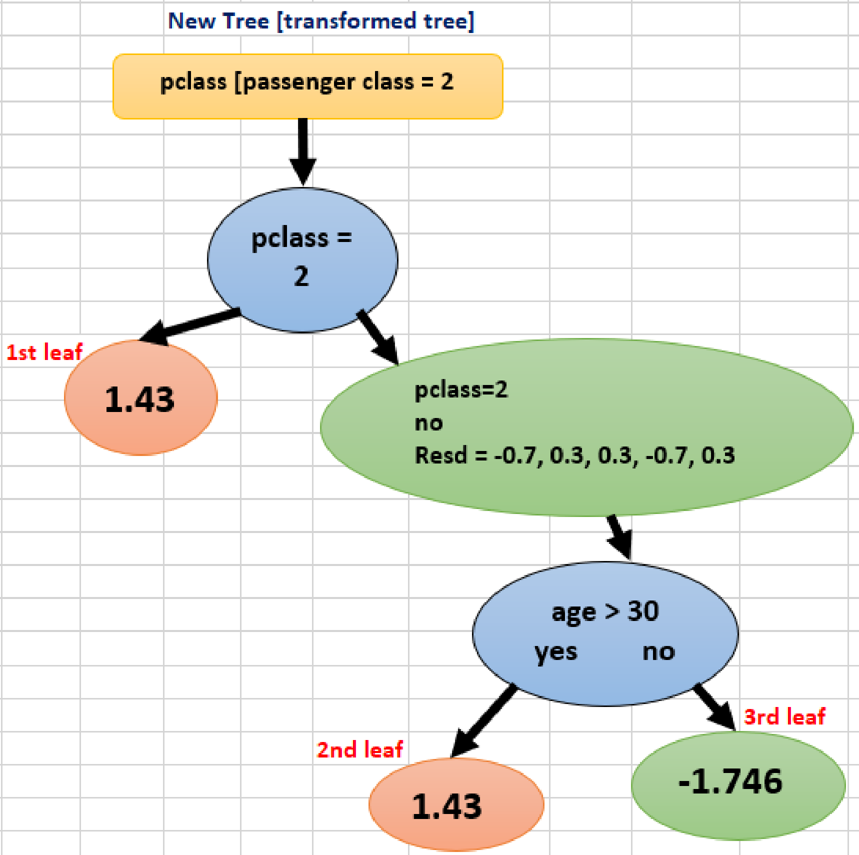
1= survived, 0= not

$$transformed\ value = \frac{\sum Residuals}{\sum[Previous\ Prob \times (1 - Previous\ Prob]}$$

$$transformed\ @\ leaf1 = \frac{0.3}{0.7 * (1 - 0.7)} = 1.43$$

$$transformed\ @\ leaf2 = \frac{0.3 + 0.3}{0.7 * (1 - 0.7) + 0.7 * (1 - 0.7)} = 1.43$$

$$transformed\ @\ leaf3 = \frac{-0.7 - 0.7 + 0.3}{0.7 * (1 - 0.7) + 0.7 * (1 - 0.7) + 0.7 * (1 - 0.7)} = -1.746$$



$$new\ log(odds) = previous\ probabilty + learning\ rate \times transformed\ value$$

	pclass	age	fare	sex	survived	pred1 (prob	resd1	log(odd)_2	odds	prob2	resd2
1	3	22	7.25	m	0	0.7	-0.7	0.5254	1.6911352	0.62841	0.07159
2	1	38	71.28	f	1	0.7	0.3	0.843	2.3233265	0.699097	0.000903
3	2	26	7.93	f	1	0.7	0.3	0.843	2.3233265	0.699097	0.000903
4	1	35	53.1	f	1	0.7	0.3	0.843	2.3233265	0.699097	0.000903
5	3	8	21.07	m	0	0.7	-0.7	0.5254	1.6911352	0.62841	0.07159
6	3	27	11.13	f	1	0.7	0.3	0.5254	1.6911352	0.62841	0.07159

1= survived, 0= not

odds	prob2
e^log(odd)_2	
1.691135167	0.62841

$$prob = \frac{e^{log(odds)}}{1+e^{log(odds)}} = \frac{odds}{1+odds}$$

e= 2.718282

Repetition of the same process will lead to improveents in probablities

Look at prob2 and see changes in estimates!

THE BEAUTIFUL THING ABOUT LEARNING IS NOBODY CAN TAKE IT AWAY FROM YOU.