20. Data Science – Machine Learning – Decision Tree

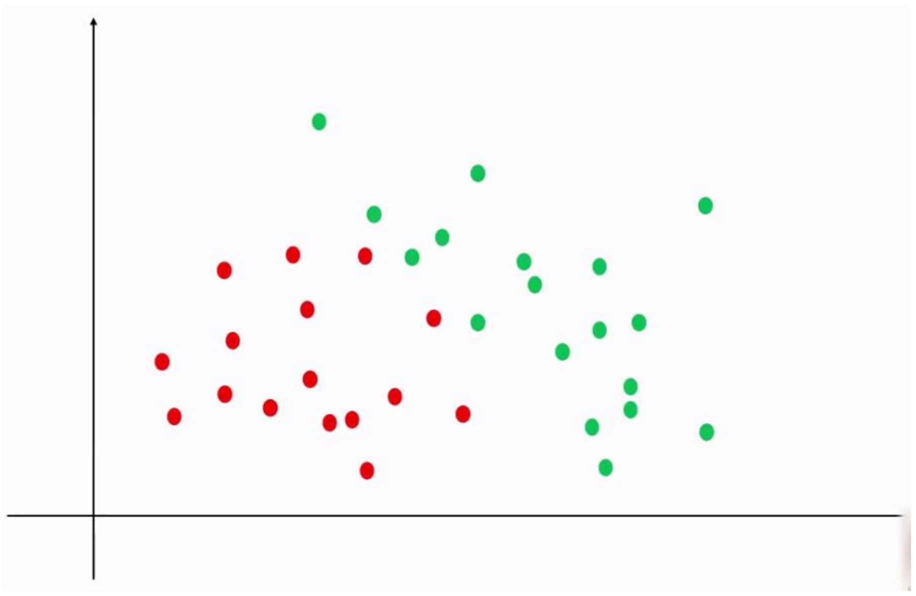
Contents

1. If dataset is like below example1 2
2. If dataset is as below example2 3
3. Decision Tree Classification Algorithm 4
3.1. Decision Node 4
3.2. Leaf nodes 4
4. CART algorithm 5
5. Why use Decision Trees? 5
6. Decision Tree Terminologies 6
6.1. Root Node 6
6.2. Leaf Node 6
6.3. Splitting 6
6.4. Branch/Sub Tree 6
6.5. Pruning 6
6.6. Parent/Child node 6
7. How does the Decision Tree algorithm Work? 7
7.1. Example 8
8. Problem 9
9. DecisionTreeClassifier class 17
9.1. fit(X_train, y_train) method 18
9.2. predict(p) method 20

20. Data Science – Machine Learning – Decision Tree

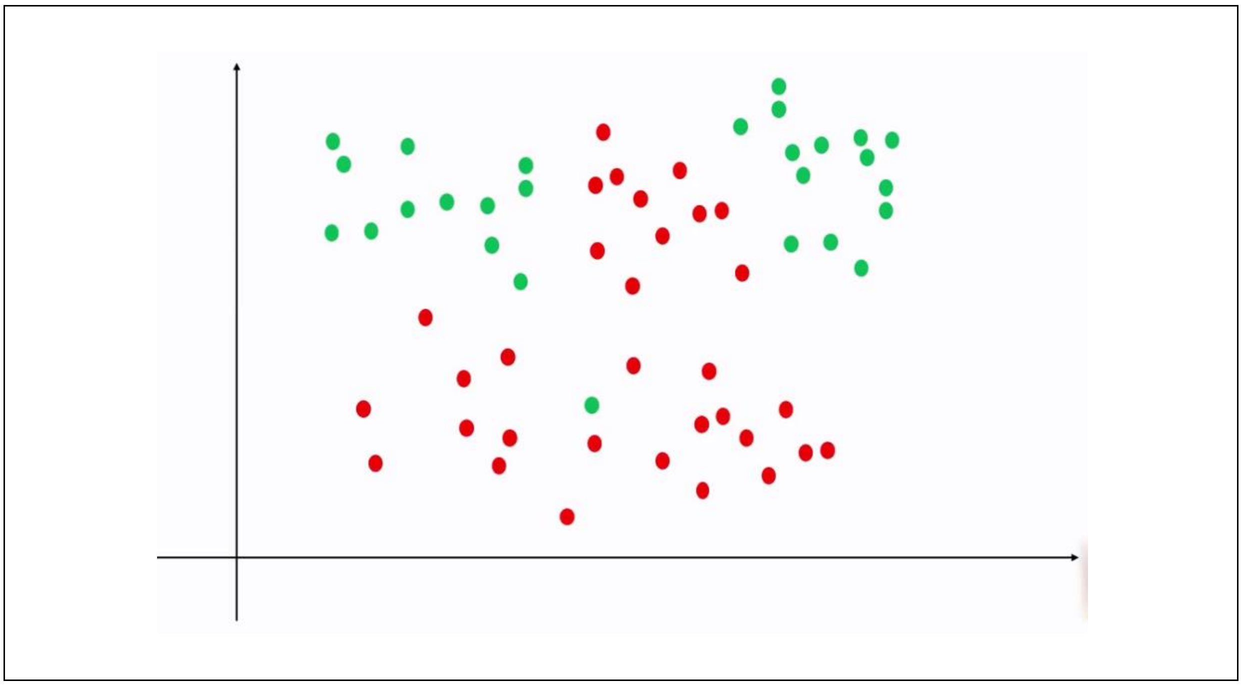
1. If dataset is like below example1

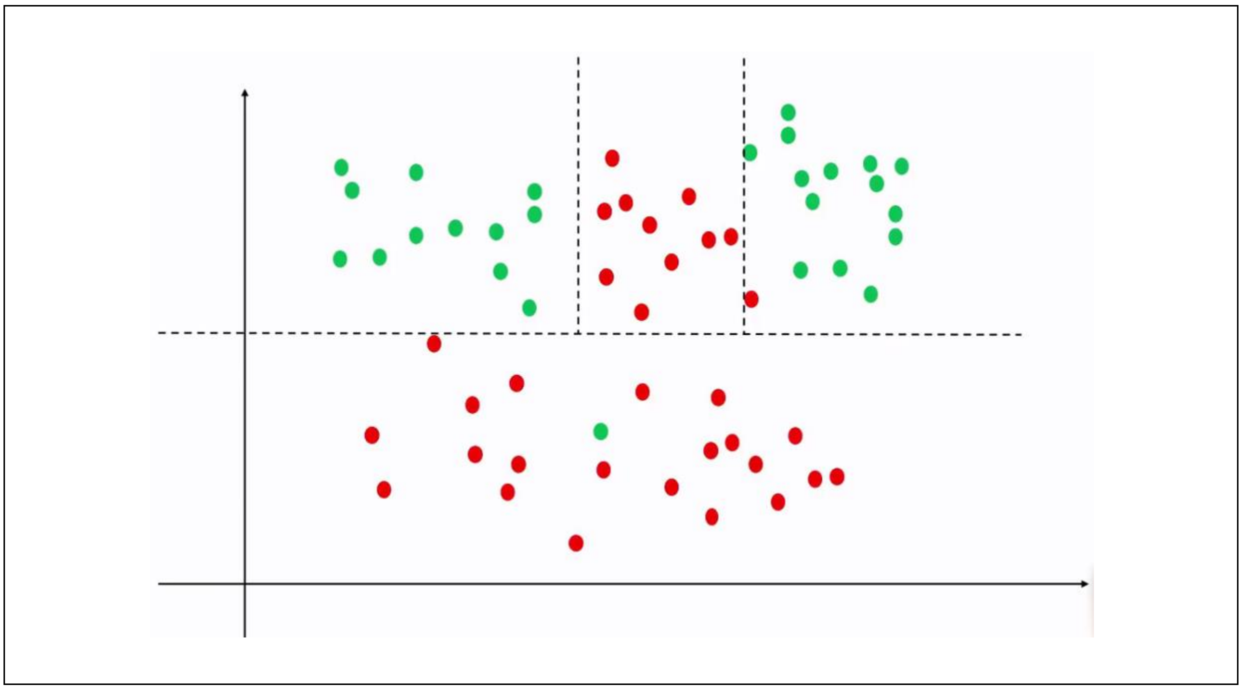
- ✓ We can draw a good separation line by using regression algorithm



2. If dataset is as below example2

- ✓ In the given below scenario, dataset is very complex.
- ✓ Then a single line may not fit for the given dataset
- ✓ Here decision tree comes into the picture





3. Decision Tree Classification Algorithm

- ✓ Decision Tree is a supervised learning technique.
- ✓ It can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
- ✓ It is called a decision tree because similar to a tree structure like it starts with the root node and expands on further branches and constructs a tree-like structure.
- ✓ In a Decision tree there are two nodes,
    - Decision Node
    - Leaf Node.

3.1. Decision Node

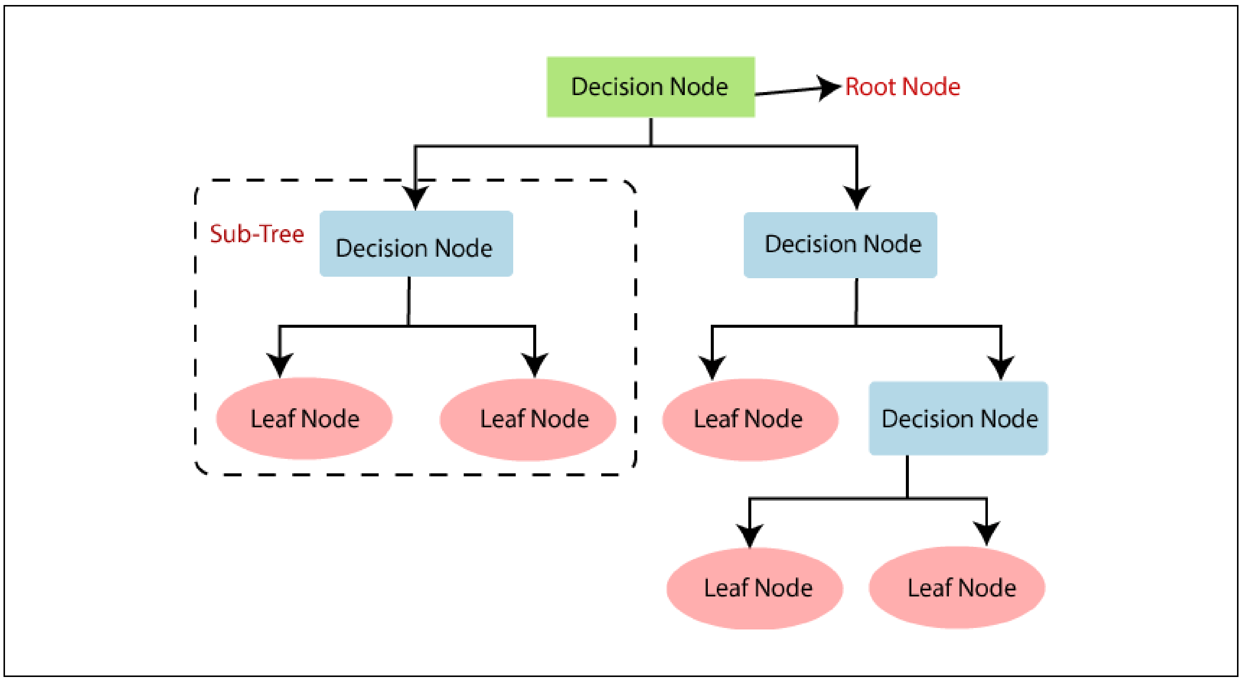
- ✓ Decision nodes are used to make any decision and have multiple branches.

3.2. Leaf nodes

- ✓ Leaf nodes are the output of those decisions and do not contain any further branches.

4. CART algorithm

- ✓ In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.
- ✓ A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.



5. Why use Decision Trees?

- ✓ Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- ✓ The logic behind the decision tree can be easily understood because it shows a tree-like structure.

6. Decision Tree Terminologies

6.1. Root Node

- ✓ Root node is from where the decision tree starts.
- ✓ It represents the entire dataset, which further gets divided into two or more homogeneous sets.

6.2. Leaf Node

- ✓ Leaf nodes are the final output node.
- ✓ The tree cannot be segregated further after getting a leaf node.

6.3. Splitting

- ✓ Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.

6.4. Branch/Sub Tree

- ✓ A tree formed by splitting the tree.

6.5. Pruning

- ✓ Pruning is the process of removing the unwanted branches from the tree.

6.6. Parent/Child node

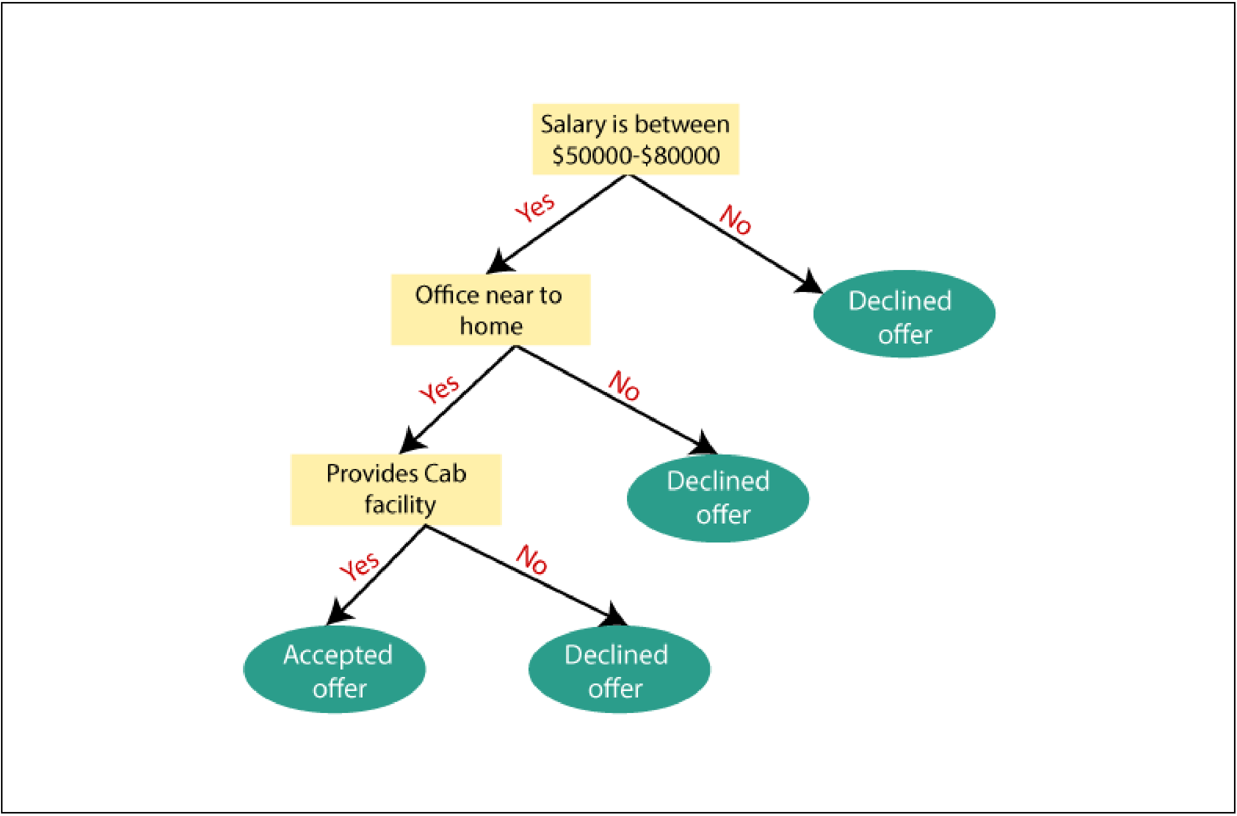
- ✓ The root node of the tree is called the parent node, and other nodes are called the child nodes.

7. How does the Decision Tree algorithm Work?

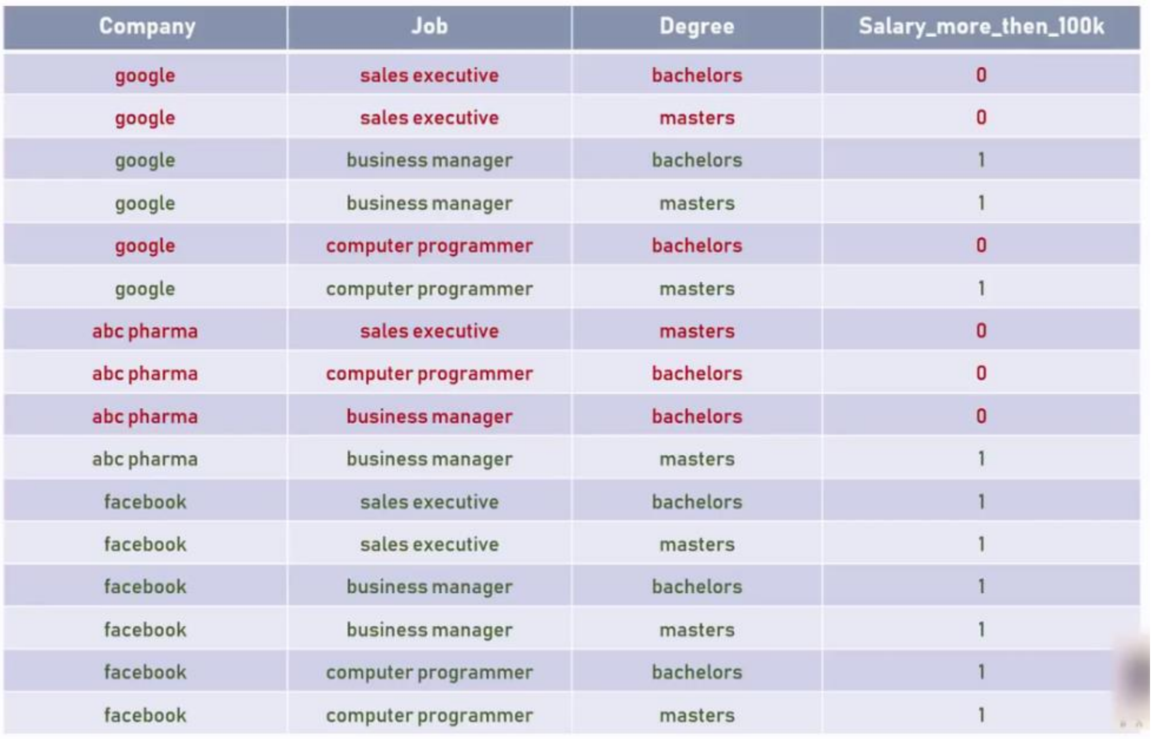
- ✓ Step-1: Begin the tree with the root node, which contains the complete dataset.
- ✓ Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- ✓ Step-3: Divide the dataset into subsets that contains possible values for the best attributes.
- ✓ Step-4: Generate the decision tree node, which contains the best attribute.
- ✓ Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

7.1. Example

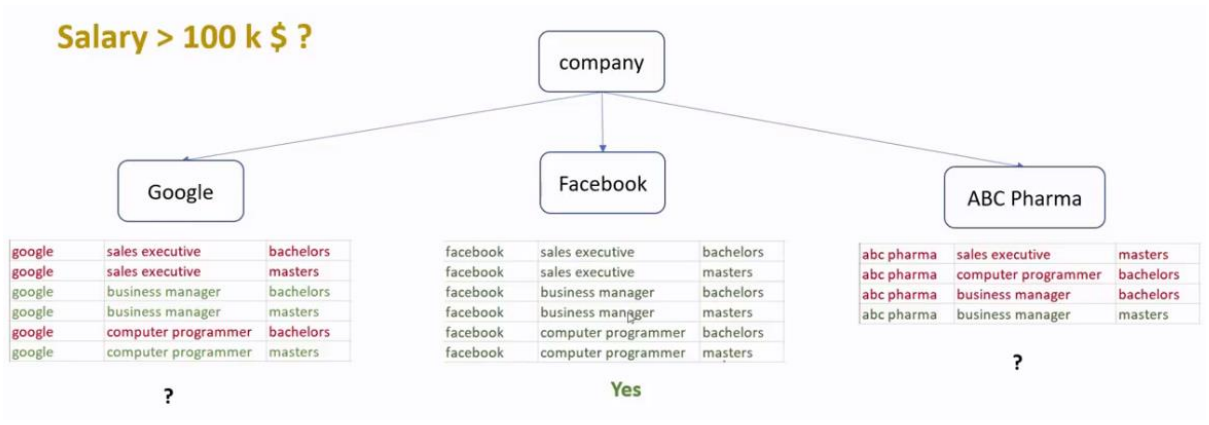
- ✓ Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not.
- ✓ So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM).
- ✓ The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels.
- ✓ The next decision node further gets split into one decision node (Cab facility) and one leaf node.
- ✓ Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer).



8. Problem



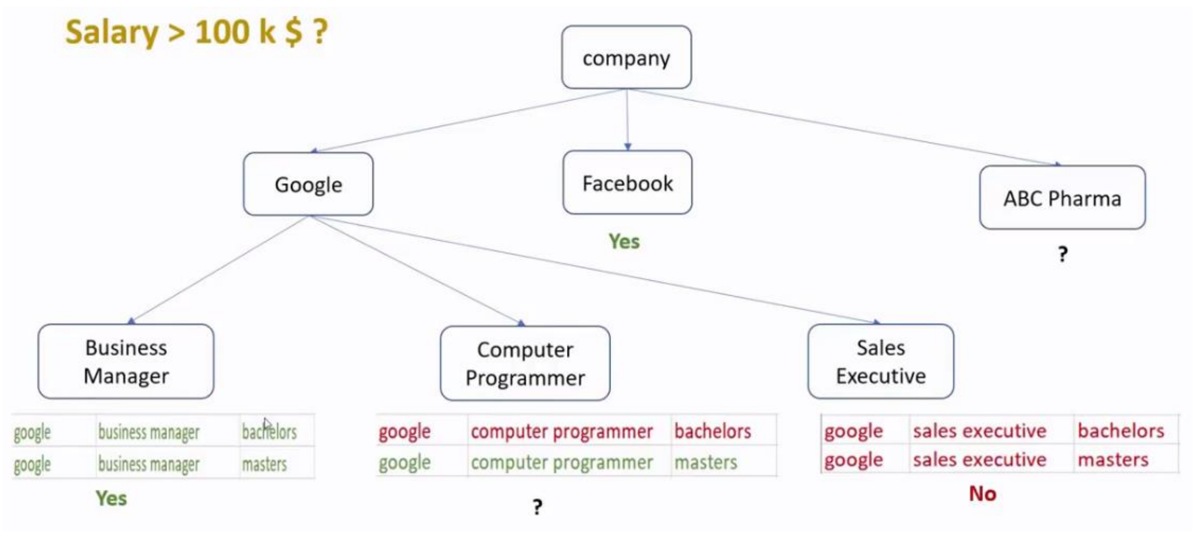
Company	Job	Degree	Salary_more_then_100k
google	sales executive	bachelors	0
google	sales executive	masters	0
google	business manager	bachelors	1
google	business manager	masters	1
google	computer programmer	bachelors	0
google	computer programmer	masters	1
abc pharma	sales executive	masters	0
abc pharma	computer programmer	bachelors	0
abc pharma	business manager	bachelors	0
abc pharma	business manager	masters	1
facebook	sales executive	bachelors	1
facebook	sales executive	masters	1
facebook	business manager	bachelors	1
facebook	business manager	masters	1
facebook	computer programmer	bachelors	1
facebook	computer programmer	masters	1

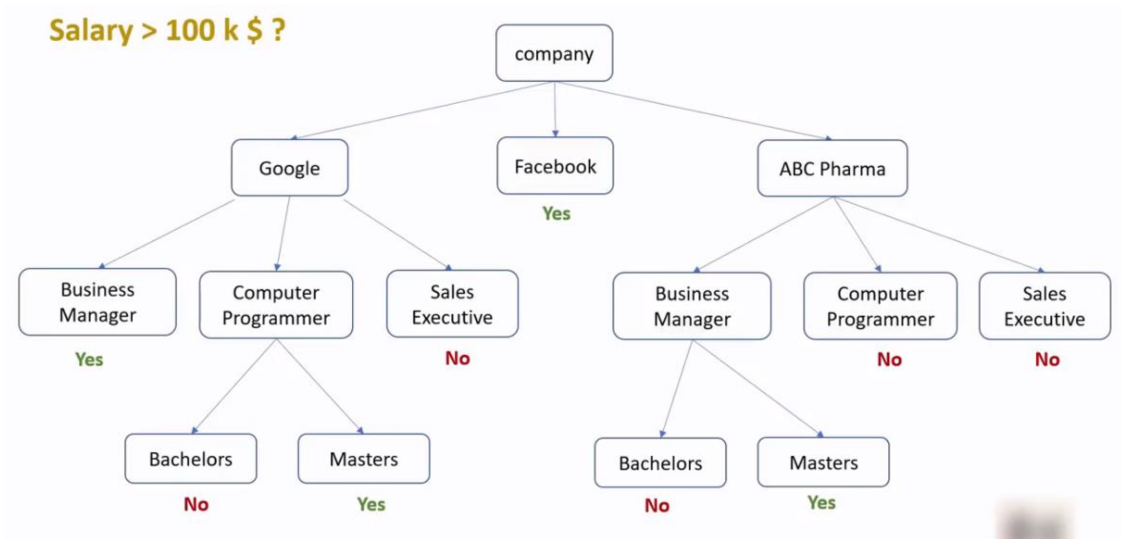


google	sales executive	bachelors
google	sales executive	masters
google	business manager	bachelors
google	business manager	masters
google	computer programmer	bachelors
google	computer programmer	masters

facebook	sales executive	bachelors
facebook	sales executive	masters
facebook	business manager	bachelors
facebook	business manager	masters
facebook	computer programmer	bachelors
facebook	computer programmer	masters

abc pharma	sales executive	masters
abc pharma	computer programmer	bachelors
abc pharma	business manager	bachelors
abc pharma	business manager	masters

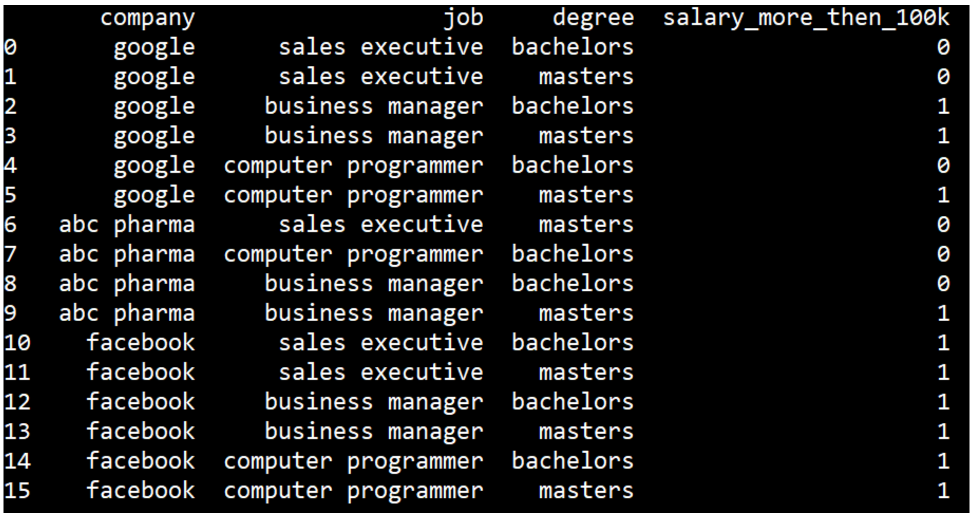

Salary > 100 k $?
company
Google
Facebook
Yes
ABC Pharma
?
Business Manager
Computer Programmer
Sales Executive
google business manager bachelors
google business manager masters
Yes
google computer programmer bachelors
google computer programmer masters
?
google sales executive bachelors
google sales executive masters
No


Salary > 100 k $?
company
Google
Facebook
Yes
ABC Pharma
Business Manager
Yes
Computer Programmer
Sales Executive
No
Bachelors
No
Masters
Yes
Business Manager
Computer Programmer
No
Sales Executive
No
Bachelors
No
Masters
Yes

Program Name	Loading dataset demo1.py

```
import pandas as pd

df = pd.read_csv("salaries.csv")
print(df)
```

Output



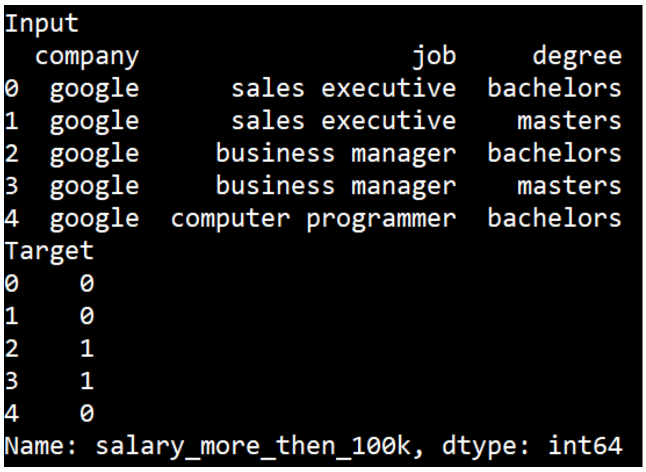
```
       company                job     degree  salary_more_then_100k
0       google      sales executive  bachelors                      0
1       google      sales executive    masters                      0
2       google     business manager  bachelors                      1
3       google     business manager    masters                      1
4       google  computer programmer  bachelors                      0
5       google  computer programmer    masters                      1
6   abc pharma      sales executive    masters                      0
7   abc pharma  computer programmer  bachelors                      0
8   abc pharma     business manager  bachelors                      0
9   abc pharma     business manager    masters                      1
10    facebook      sales executive  bachelors                      1
11    facebook      sales executive    masters                      1
12    facebook     business manager  bachelors                      1
13    facebook     business manager    masters                      1
14    facebook  computer programmer  bachelors                      1
15    facebook  computer programmer    masters                      1
```

Program Name

Preparing input and target
demo2.py

```
import pandas as pd

df = pd.read_csv("salaries.csv")

inputs = df.drop('salary_more_then_100k', axis = 'columns')
target = df['salary_more_then_100k']

print("Input")
print(inputs.head())

print("Target")
print(target.head())
```

Output



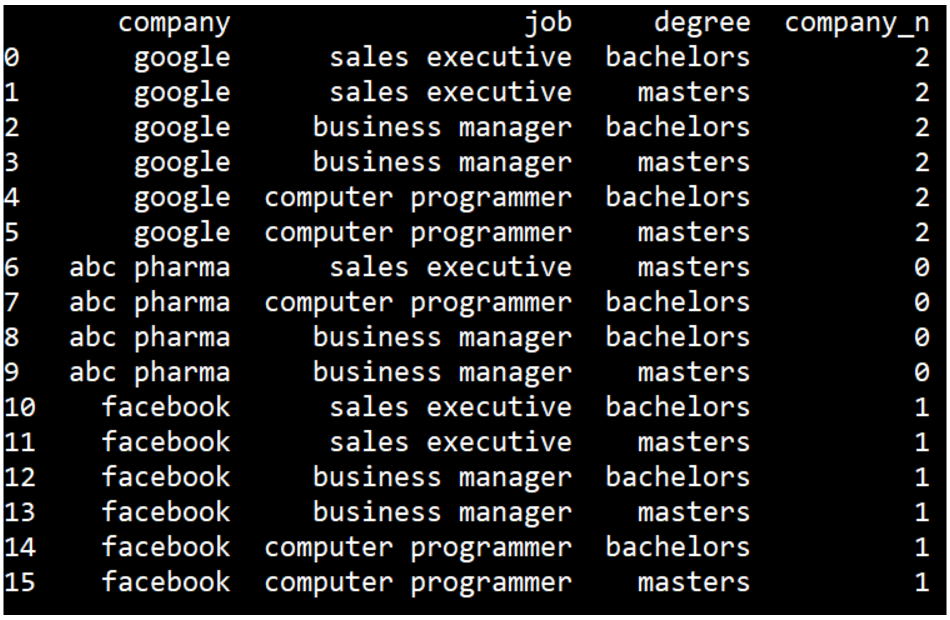
```
Input
  company                 job     degree
0  google     sales executive  bachelors
1  google     sales executive    masters
2  google    business manager  bachelors
3  google    business manager    masters
4  google  computer programmer  bachelors
Target
0    0
1    0
2    1
3    1
4    0
Name: salary_more_then_100k, dtype: int64
```

Program Name

Transforming input
demo3.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("salaries.csv")

inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])

print(inputs)
```

Output



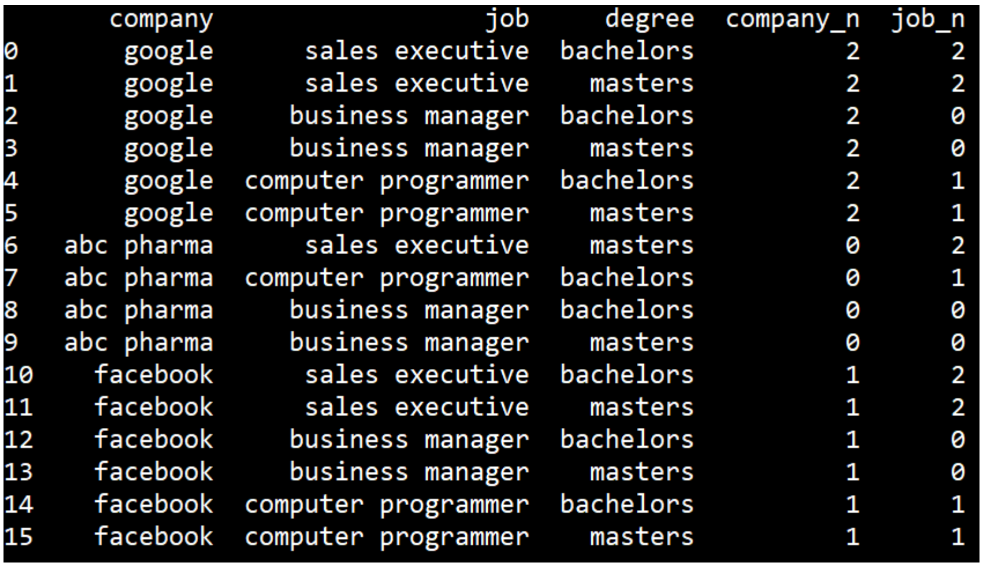
```
       company                  job     degree  company_n
0       google      sales executive  bachelors          2
1       google      sales executive    masters          2
2       google     business manager  bachelors          2
3       google     business manager    masters          2
4       google  computer programmer  bachelors          2
5       google  computer programmer    masters          2
6   abc pharma      sales executive    masters          0
7   abc pharma  computer programmer  bachelors          0
8   abc pharma     business manager  bachelors          0
9   abc pharma     business manager    masters          0
10    facebook      sales executive  bachelors          1
11    facebook      sales executive    masters          1
12    facebook     business manager  bachelors          1
13    facebook     business manager    masters          1
14    facebook  computer programmer  bachelors          1
15    facebook  computer programmer    masters          1
```

Program Name

Transforming input
demo4.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("salaries.csv")

inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])

print(inputs)
```

Output



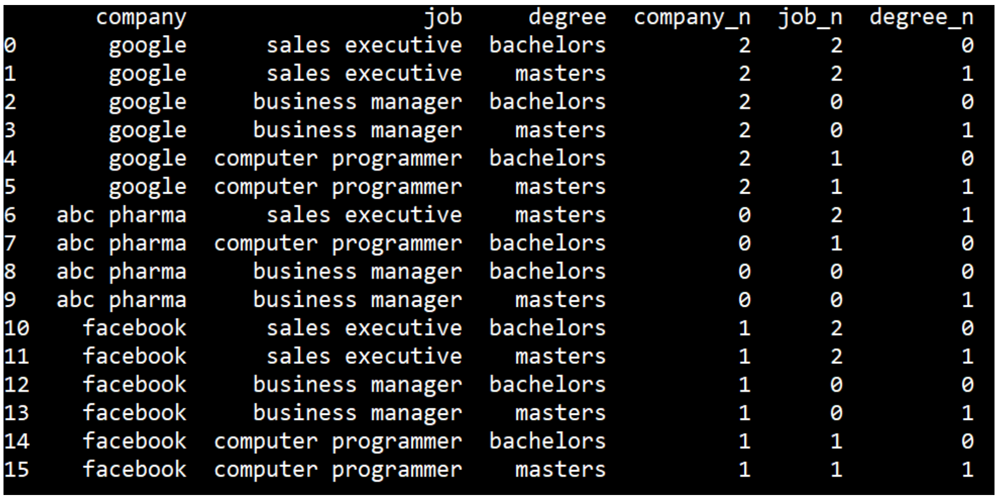
```
       company                 job     degree  company_n  job_n
0       google     sales executive  bachelors          2      2
1       google     sales executive    masters          2      2
2       google    business manager  bachelors          2      0
3       google    business manager    masters          2      0
4       google  computer programmer  bachelors          2      1
5       google  computer programmer    masters          2      1
6   abc pharma     sales executive    masters          0      2
7   abc pharma  computer programmer  bachelors          0      1
8   abc pharma    business manager  bachelors          0      0
9   abc pharma    business manager    masters          0      0
10    facebook     sales executive  bachelors          1      2
11    facebook     sales executive    masters          1      2
12    facebook    business manager  bachelors          1      0
13    facebook    business manager    masters          1      0
14    facebook  computer programmer  bachelors          1      1
15    facebook  computer programmer    masters          1      1
```

Program Name

Transforming input
demo5.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

print(inputs)
```

Output



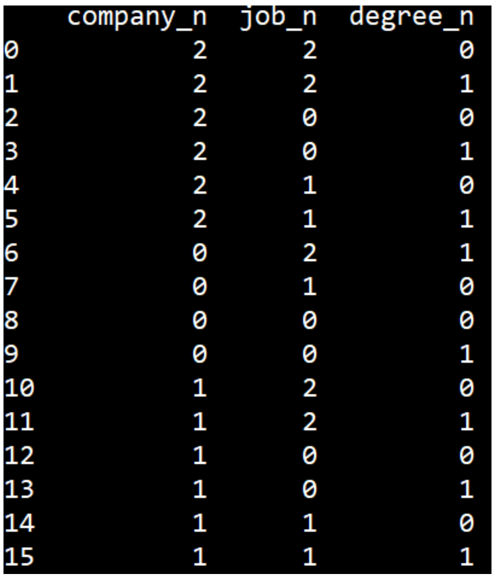
```
       company                 job     degree  company_n  job_n  degree_n
0       google     sales executive  bachelors          2      2         0
1       google     sales executive    masters          2      2         1
2       google    business manager  bachelors          2      0         0
3       google    business manager    masters          2      0         1
4       google  computer programmer  bachelors         2      1         0
5       google  computer programmer    masters         2      1         1
6   abc pharma     sales executive    masters          0      2         1
7   abc pharma  computer programmer  bachelors         0      1         0
8   abc pharma    business manager  bachelors          0      0         0
9   abc pharma    business manager    masters          0      0         1
10    facebook     sales executive  bachelors          1      2         0
11    facebook     sales executive    masters          1      2         1
12    facebook    business manager  bachelors          1      0         0
13    facebook    business manager    masters          1      0         1
14    facebook  computer programmer  bachelors         1      1         0
15    facebook  computer programmer    masters         1      1         1
```

Program Name

Transforming input
demo6.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

inputs_n = inputs.drop(['company', 'job', 'degree'], axis='columns')

print(inputs_n)
```

Output



```
    company_n  job_n  degree_n
0           2      2         0
1           2      2         1
2           2      0         0
3           2      0         1
4           2      1         0
5           2      1         1
6           0      2         1
7           0      1         0
8           0      0         0
9           0      0         1
10          1      2         0
11          1      2         1
12          1      0         0
13          1      0         1
14          1      1         0
15          1      1         1
```

9. DecisionTreeClassifier class

- ✓ **DecisionTreeClassifier** is predefined class in **sklearn.tree** package
- ✓ We need to import this class from **sklearn.tree** package
- ✓ Once we imported then we need to **create an object** to **DecisionTreeClassifier** class.

Program Name: Model creation
demo7.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

inputs_n = inputs.drop(['company', 'job', 'degree'], axis='columns')

model = DecisionTreeClassifier()
print("DecisionTreeClassifier object created")
```

Output

```
DecisionTreeClassifier object created
```

9.1. fit(X_train, y_train) method

- ✓ fit(X_train, y_train) is predefined method in DecisionTreeClassifier class.
- ✓ We should access this method by using DecisionTreeClassifier object only.
- ✓ By using this method we need to train the model.

Program Name: Model creation
demo8.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

inputs_n = inputs.drop(['company', 'job', 'degree'], axis='columns')

model = DecisionTreeClassifier()
model.fit(inputs_n.values, target)

print("Model got trained")
```

Output

```
Model got trained
```

Program Name

Model score
demo9.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

inputs_n = inputs.drop(['company', 'job', 'degree'], axis='columns')

print("Model got trained")
model = DecisionTreeClassifier()
model.fit(inputs_n.values, target)

print(model.score(inputs_n, target))
```

Output

```
1.0
```

9.2. predict(p) method

- ✓ predict(p) is predefined method in DecisionTreeClassifier class.
- ✓ We should access this method by using DecisionTreeClassifier object only.
- ✓ By using this method we can predict the results.

Program Name Model prediction
demo10.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

inputs_n = inputs.drop(['company', 'job', 'degree'], axis='columns')

print("Model got trained")
model = DecisionTreeClassifier()
model.fit(inputs_n.values, target)

print(model.predict([[2, 1, 0]]))
```

Output

```
[0]
```

Program Name Model prediction
demo11.py

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier

df = pd.read_csv("salaries.csv")
inputs = df.drop('salary_more_then_100k',axis='columns')
target = df['salary_more_then_100k']

le_company = LabelEncoder()

inputs['company_n'] =
le_company.fit_transform(inputs['company'])
inputs['job_n'] = le_company.fit_transform(inputs['job'])
inputs['degree_n'] = le_company.fit_transform(inputs['degree'])

inputs_n = inputs.drop(['company', 'job', 'degree'], axis='columns')

print("Model got trained")
model = DecisionTreeClassifier()
model.fit(inputs_n.values, target)

print(model.predict([[2, 1, 1]]))
```

Output

```
[1]
```