* **MINI-PROJECT REPORT**

**Employee Attrition Prediction Using Decision Trees**



**NAME: Mosuri Janaki Srivalli**

**PS NO: 40027796**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No.** | **CONTENT** | **PAGE No.** |
| 1 | Problem Statement | 3 |
| 2 | Dataset | 3 |
| 3 | Algorithm/Model | 4 |
| 4 | Experiment/Implementation | 5-8 |
| 5 | Results | 9 |
| 6 | Conclusion | 9 |
| 7 | References | 9 |

**1.Problem Statement**

Employee attrition is an extremely important and relevant business problem all organizations face. It will give a huge advantage to companies and their HR teams, if they can predict with certain level of confidence, which employees are most likely to leave. They are either replaced by new employees, or sometimes the role could remain vacant or closed altogether. We have an employee attrition data of a big Organization.

**Introduction:**

In this project,we strives to use machine learning principles to predict employee attrition,provide managerial insights to prevent attrition,and finally rule out and present the factors that lead to attrition.I used to like Decision trees on the given data set.It represents **fictitious/fake data** on terminations. For each of 10 years it shows employees that are active and those that terminated. The intent is to see if individual terminations can be predicted from the data provided. The thing to be predicted is status of active or terminated.

**2. Dataset**

In the given dataset there are different parameters.Based on that parameters,we compared between that parameters we will get results. It has a total of 49K records and 9 parameters in the dataset. Itcontains EmployeeID,orighiredate\_key,terminationdate\_key,age,length\_of\_service,job\_title,gender,termreason\_desc,termtype\_desc,STATUS\_YEAR,STATUS.This dataset predict the value from status parameter.

Graphical user interface

Description automatically generated with medium confidence

**Top 5 observations From the terminationdata dataset**

**3.Algorithm/Model**

In this Mini Project I am going to use supervised machine learning

algorithm (Decision Trees)and predict the accurate results from this algorithm.

* **Decision Tree Algorithm:**
* Decision Tree is a **Supervised learning technique**that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems.
* It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome**.**
* In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* 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.
* Below diagram explains the general structure of a decision tree:

**Implementation of Decision Tree Algorithm:**

* **Data Pre-processing step**
* **Fitting a Decision-Tree algorithm to the Training set**
* **Predicting the test result**
* **Test accuracy of the result(Creation of Confusion matrix)**

**4.Experiment/Implementation**

* **Step-1: Data Pre-Processing**

**Importing Libraries**

import numpy as np

import pandas as pd

import matplotlib. pyplot as plot

import seaborn as sns

**Importing Data file and displaying what columns are available in the dataset.**

dataset=pd.read\_cs('terminationdata.csv')

dataset.columns

**Output:**

Text

Description automatically generated with medium confidence

**Data Preprocessing:**

from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()

dataset['job\_title'] = le.fit\_transform(dataset['job\_title'])

dataset['gender'] = le.fit\_transform(dataset['gender'])

dataset['termreason\_desc'] = le.fit\_transform(dataset['termreason\_desc'])

dataset['termtype\_desc'] = le.fit\_transform(dataset['termtype\_desc'])

dataset['STATUS\_YEAR'] = le.fit\_transform(dataset['STATUS\_YEAR'])

dataset

**Output:**

Table

Description automatically generated with medium confidence

* **Step-2:Fitting a Decision-Tree algorithm to the Training set**

Now we will fit the model to the training set. For this, we will import the DecisionTreeClassifier class from sklearn.tree library. Below is the code for it:

**#Fitting Decision Tree classifier to the training set**

1. From sklearn.tree **import** DecisionTreeClassifier
2. classifier= DecisionTreeClassifier( random\_state=0)
3. classifier.fit(x\_train, y\_train)

In the above code, we have created a classifier object, in which we have passed two main parameters;

**random\_state=0":** For generating the random states.

* **Plotting the graph:**

sns.countplot(x='STATUS', data=dataset)

from sklearn.model\_selection import train\_test\_split

Chart, bar chart

Description automatically generated

* **Heatmap :**

import matplotlib as plt

corr = dataset.corr()

plt.pyplot.subplots(figsize=(15,10))

sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, annot=True, cmap=sns.diverging\_palette(220, 20, as\_cmap=True))

Chart, table, treemap chart

Description automatically generated

* **Step-3. Predicting the test result**

Now we will predict the test set result. We will create a new prediction vector **y\_pred.** Below is the code for it:

#Predicting the test set result

1. y\_pred= classifier.predict(x\_test)
2. y\_pred

**Output:**

In the below output image, the predicted output and real test output are given. We can clearly see that there are some values in the prediction vector, which are different from the real vector values. These are prediction errors.

* **Step-4. Test accuracy of the result (Creation of Confusion matrix)**

In the above output, we have seen that there were some incorrect predictions, so if we want to know the number of correct and incorrect predictions, we need to use the confusion matrix. Below is the code for it:

#Creating the Confusion matrix

1. from sklearn.metrics **import** confusion\_matrix
2. cm= confusion\_matrix(y\_test, y\_pred)
3. cm

**Output:**

In the above output image, we can see the confusion matrix, which has **0+0=0 incorrect predictions** and **12034+380=12414 correct predictions. Therefore, we can say that compared to other classification models, the Decision Tree classifier made a good prediction.**

**Result**

After implementation part I got result for Employee Attrition.My accuracy score for this Employee Attrition is given below.



**Conclusion**

For this project all are used different classifications,but I used Decision Tree classification because my dataset parameters based on some conditions that’s why I choose decision tree classification.

The decision tree algorithm model is efficient enough to produce the outcome and test data, as can be seen from its accuracy score of 1.0. This enables us to categorise employee attrition data based on their reasons.It overfits the data.It shows no incorrect values.

We gave user input values for this project it takes highest status value means,In my dataset I predict the values from STATUS parameter.In that Active or terminated are there.From the above given data so many employees are in active state.Thats why its overfits the data and gives output value as active.

**References:**

* <https://www.kaggle.com/datasets/HRAnalyticRepository/employee-attrition-data>
* <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>