ENGR_ALDA_FALL2021 HOMEWORK 2

Homework Team 12 - HW12

Github Repository Link

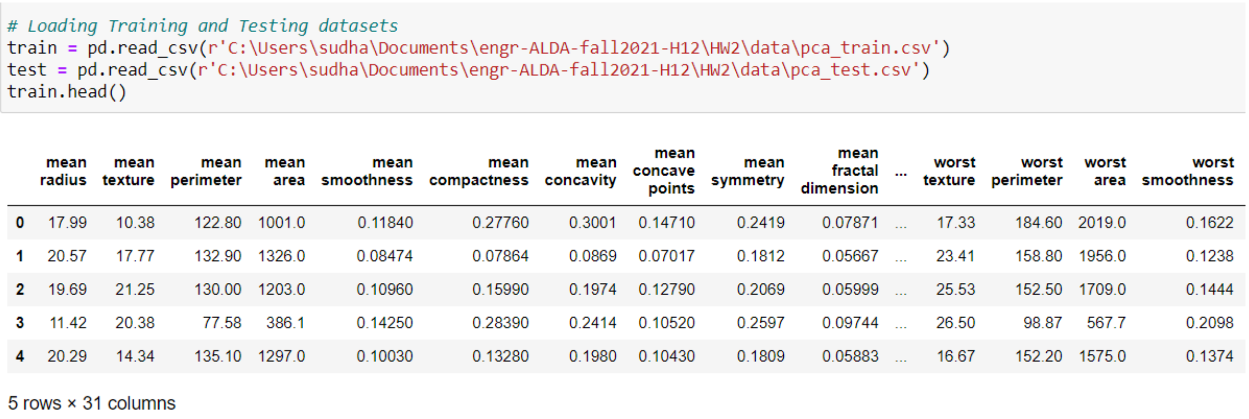
Sudharsan Janardhanan - sjanard
Sriram Sudharsan - ssudhar
Pradyumna Vemuri - pvemuri

---

1. (40 points) [PCA] [John Wesley Hostetter & Ge Gao] PCA is an unsupervised learning algorithm that uses an orthogonal transformation to convert data to new dimensions. In this problem, you will perform a PCA on the provided training dataset (\pca train.csv") and the testing dataset (\pca test.csv"). In both datasets, each row represents a data point or sample and the last column \Class" is a feature that indicates a class for each sample. The rest of the columns are input features. Write code in Python to perform the following tasks. You can use Numpy, Pandas, Matplotlib, SciPy, and Sklearn to solve this problem. Please report your output and relevant code in the document file, and also include your code le (ends with .py) in your GitHub repository.

Source code for Question 1 can be found in the Python file 1.py in the following Github repository under the H2 directory.

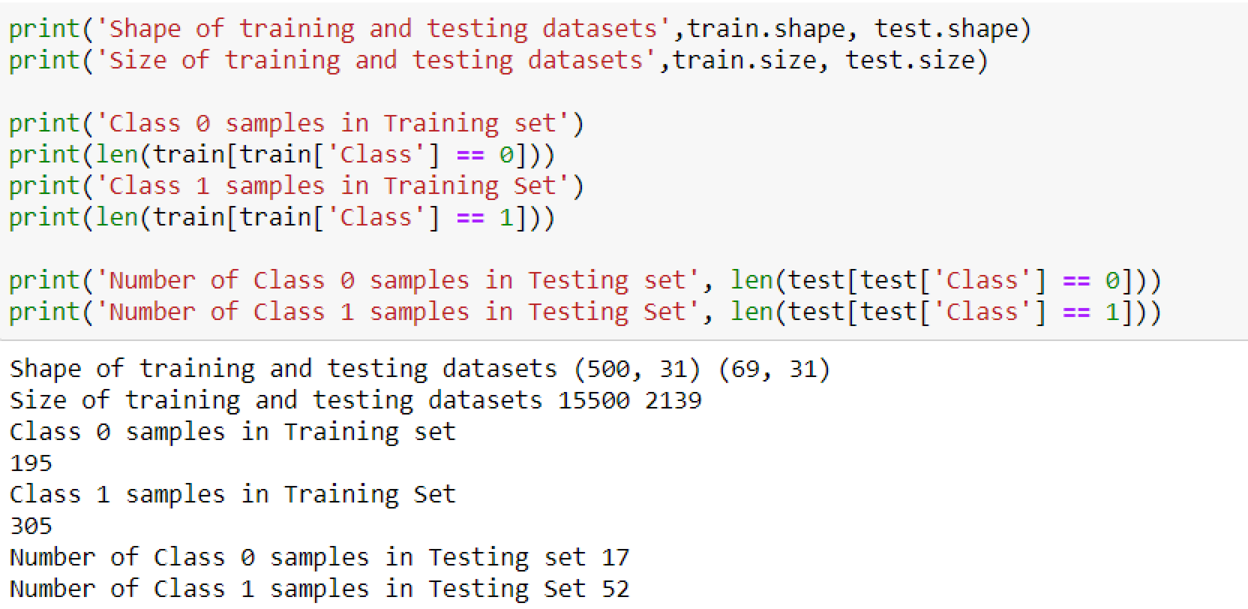
(a) (2 points) Load the data. Report the size of the training and testing sets. How many Class (1) and Class (0) samples are in the training set and the testing set, respectively?



```
# Loading Training and Testing datasets
train = pd.read_csv(r'C:\Users\sudha\Documents\engr-ALDA-fall2021-H12\HW2\data\pca_train.csv')
test = pd.read_csv(r'C:\Users\sudha\Documents\engr-ALDA-fall2021-H12\HW2\data\pca_test.csv')
train.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374

5 rows × 31 columns

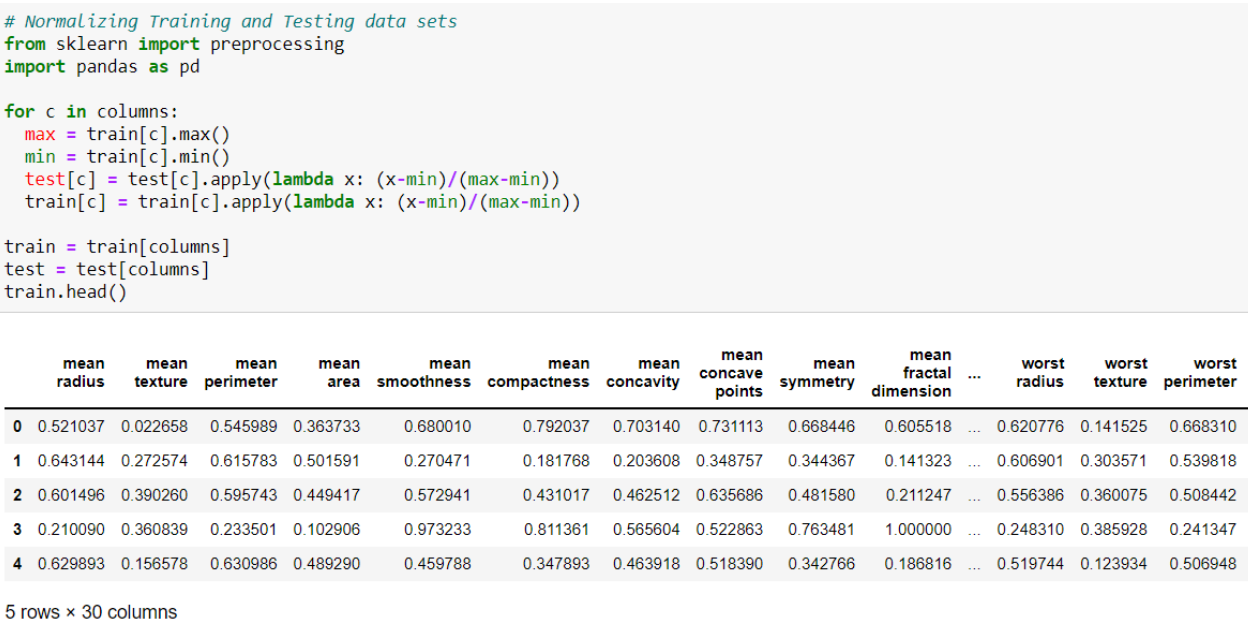


```
print('Shape of training and testing datasets',train.shape, test.shape)
print('Size of training and testing datasets',train.size, test.size)

print('Class 0 samples in Training set')
print(len(train[train['Class'] == 0]))
print('Class 1 samples in Training Set')
print(len(train[train['Class'] == 1]))

print('Number of Class 0 samples in Testing set', len(test[test['Class'] == 0]))
print('Number of Class 1 samples in Testing Set', len(test[test['Class'] == 1]))
```

```
Shape of training and testing datasets (500, 31) (69, 31)
Size of training and testing datasets 15500 2139
Class 0 samples in Training set
195
Class 1 samples in Training Set
305
Number of Class 0 samples in Testing set 17
Number of Class 1 samples in Testing Set 52
```

(b) (18 points) Preprocessing Data-Normalization: Please run normalization on all input features in both the training and testing datasets to obtain the normalized training and the normalized testing datasets. (Hint: you need to use the min/max of each column in the training dataset to normalize the testing dataset, and do NOT normalize the output “Class" of data.)

Normalized Training dataset



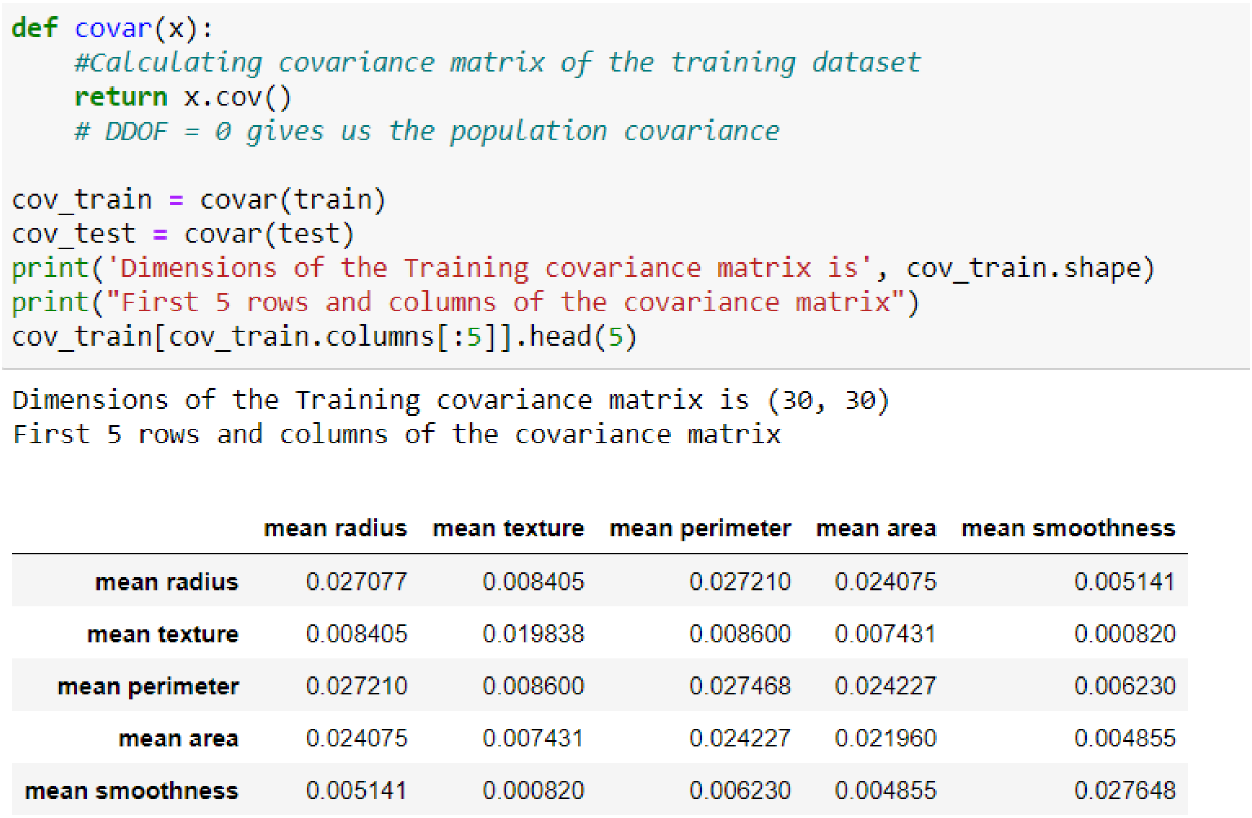
```
# Normalizing Training and Testing data sets
from sklearn import preprocessing
import pandas as pd

for c in columns:
  max = train[c].max()
  min = train[c].min()
  test[c] = test[c].apply(lambda x: (x-min)/(max-min))
  train[c] = train[c].apply(lambda x: (x-min)/(max-min))

train = train[columns]
test = test[columns]
train.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst radius	worst texture	worst perimeter
0	0.521037	0.022658	0.545989	0.363733	0.680010	0.792037	0.703140	0.731113	0.668446	0.605518	...	0.620776	0.141525	0.668310
1	0.643144	0.272574	0.615783	0.501591	0.270471	0.181768	0.203608	0.348757	0.344367	0.141323	...	0.606901	0.303571	0.539818
2	0.601496	0.390260	0.595743	0.449417	0.572941	0.431017	0.462512	0.635686	0.481580	0.211247	...	0.556386	0.360075	0.508442
3	0.210090	0.360839	0.233501	0.102906	0.973233	0.811361	0.565604	0.522863	0.763481	1.000000	...	0.248310	0.385928	0.241347
4	0.629893	0.156578	0.630986	0.489290	0.459788	0.347893	0.463918	0.518390	0.342766	0.186816	...	0.519744	0.123934	0.506948

5 rows × 30 columns

i. (2 points) Calculate the covariance matrix of the NEW training dataset. Please 1) specify the dimension of the resulted covariance matrix and 2) given the space

limitation, please report the first 5 * 5 of the covariance matrix, that is, only reporting the first five rows and the first five columns of the entire covariance Matrix.

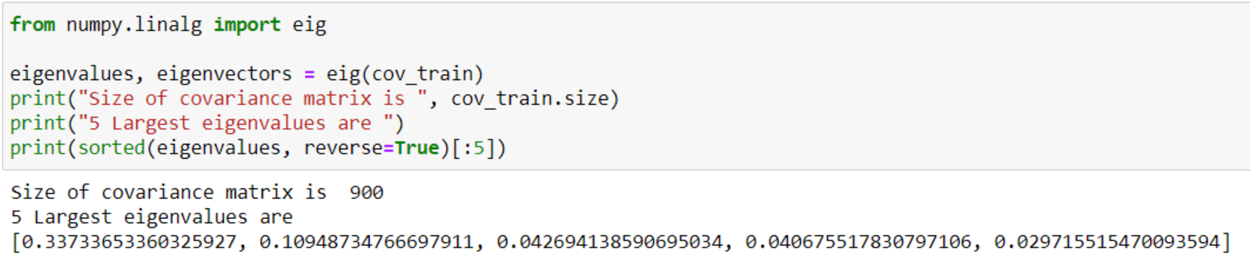


```
def covar(x):
    #Calculating covariance matrix of the training dataset
    return x.cov()
    # DDOF = 0 gives us the population covariance

cov_train = covar(train)
cov_test = covar(test)
print('Dimensions of the Training covariance matrix is', cov_train.shape)
print("First 5 rows and columns of the covariance matrix")
cov_train[cov_train.columns[:5]].head(5)
```

```
Dimensions of the Training covariance matrix is (30, 30)
First 5 rows and columns of the covariance matrix
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness
mean radius	0.027077	0.008405	0.027210	0.024075	0.005141
mean texture	0.008405	0.019838	0.008600	0.007431	0.000820
mean perimeter	0.027210	0.008600	0.027468	0.024227	0.006230
mean area	0.024075	0.007431	0.024227	0.021960	0.004855
mean smoothness	0.005141	0.000820	0.006230	0.004855	0.027648

ii. (2 points) Calculate the eigenvalues and the eigenvectors based on the entire covariance matrix in (i) above. Report the size of the covariance matrix and the 5 largest eigenvalues.

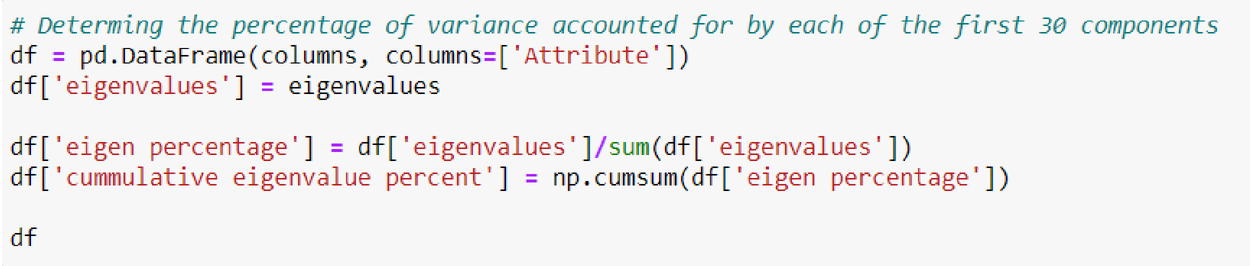


```
from numpy.linalg import eig

eigenvalues, eigenvectors = eig(cov_train)
print("Size of covariance matrix is ", cov_train.size)
print("5 Largest eigenvalues are ")
print(sorted(eigenvalues, reverse=True)[:5])
```

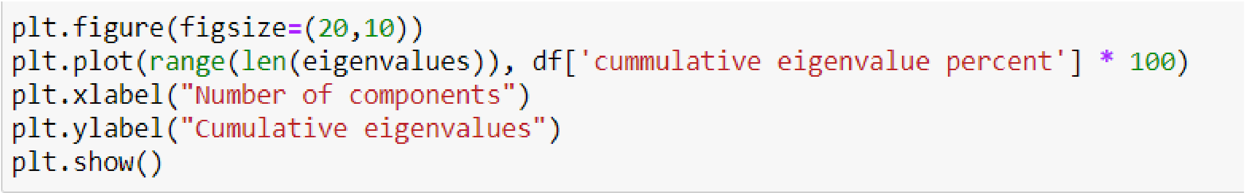
```
Size of covariance matrix is  900
5 Largest eigenvalues are
[0.33733653360325927, 0.10948734766697911, 0.042694138590695034, 0.040675517830797106, 0.029715515470093594]
```

iii. (1 point) Display the eigenvalues using a bar graph or a plot, and choose a reasonable number(s) of eigenvectors. Justify your answer.

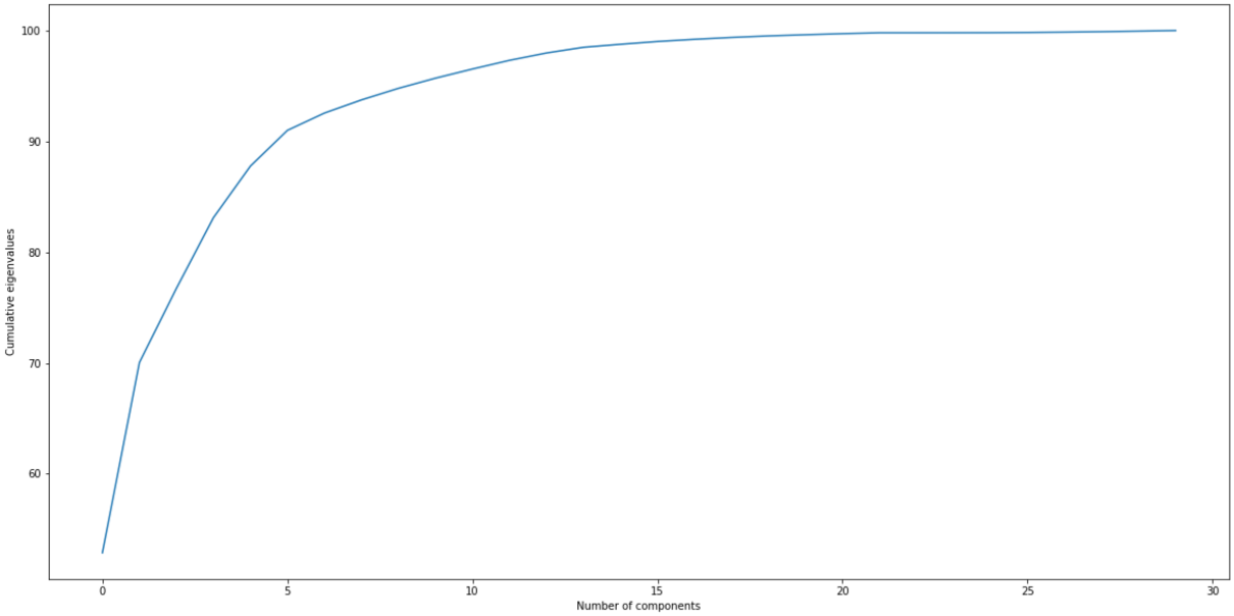
Upon analyzing the scree plot, we see that the magnitude of the slope of the curve decreases significantly after the 7th eigenvector. After the 7th eigenvector, the curve straightens out implying that the first 7 eigenvectors encode most of the important information required for PCA. (Note: The plot starts with index = 0 so the 6th eigenvector is in fact the 7th).



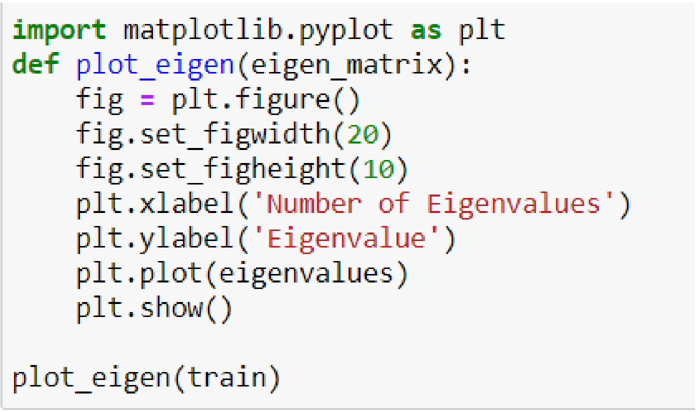
```
# Determing the percentage of variance accounted for by each of the first 30 components
df = pd.DataFrame(columns, columns=['Attribute'])
df['eigenvalues'] = eigenvalues

df['eigen percentage'] = df['eigenvalues']/sum(df['eigenvalues'])
df['cummulative eigenvalue percent'] = np.cumsum(df['eigen percentage'])

df
```



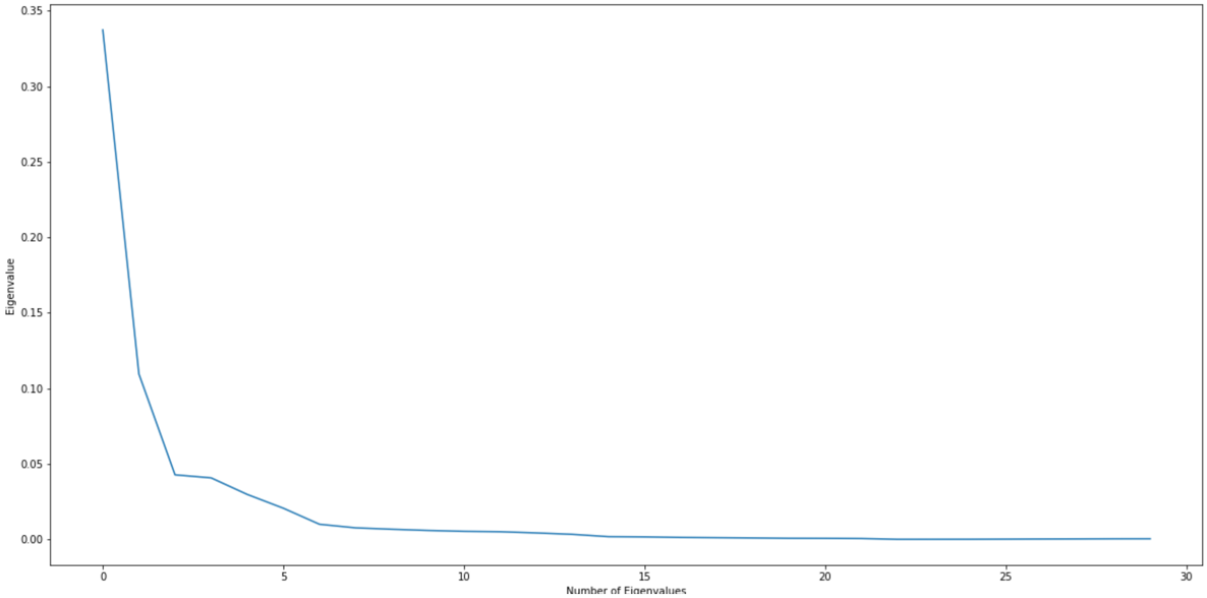
```
plt.figure(figsize=(20,10))
plt.plot(range(len(eigenvalues)), df['cummulative eigenvalue percent'] * 100)
plt.xlabel("Number of components")
plt.ylabel("Cumulative eigenvalues")
plt.show()
```



As we can see from the plot above, the cumulative eigenvalue percentage for the 7th eigenvector encodes 92.5% of the information. The line graph below plots the Eigenvalues with their count.



```
import matplotlib.pyplot as plt
def plot_eigen(eigen_matrix):
    fig = plt.figure()
    fig.set_figwidth(20)
    fig.set_figheight(10)
    plt.xlabel('Number of Eigenvalues')
    plt.ylabel('Eigenvalue')
    plt.plot(eigenvalues)
    plt.show()

plot_eigen(train)
```

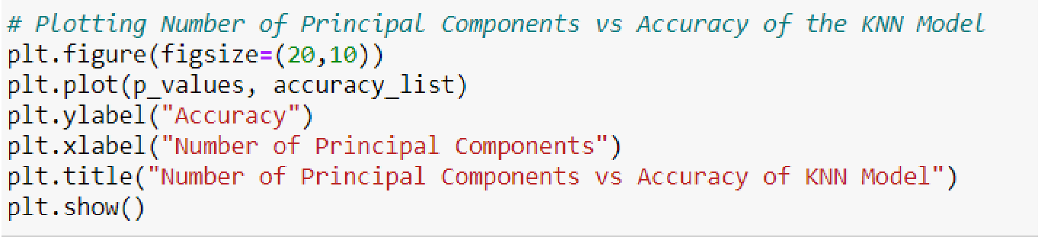


iv. (13 points) Next, you will combine PCA with a K-nearest neighbor (KNN) classifier. More specifically, PCA will be applied to reduce the dimensionality of data by transforming the original data into p (p 30) principal components; and then KNN (K = 5, Euclidean distance as distance metric) will be employed to the p principal components for classification.

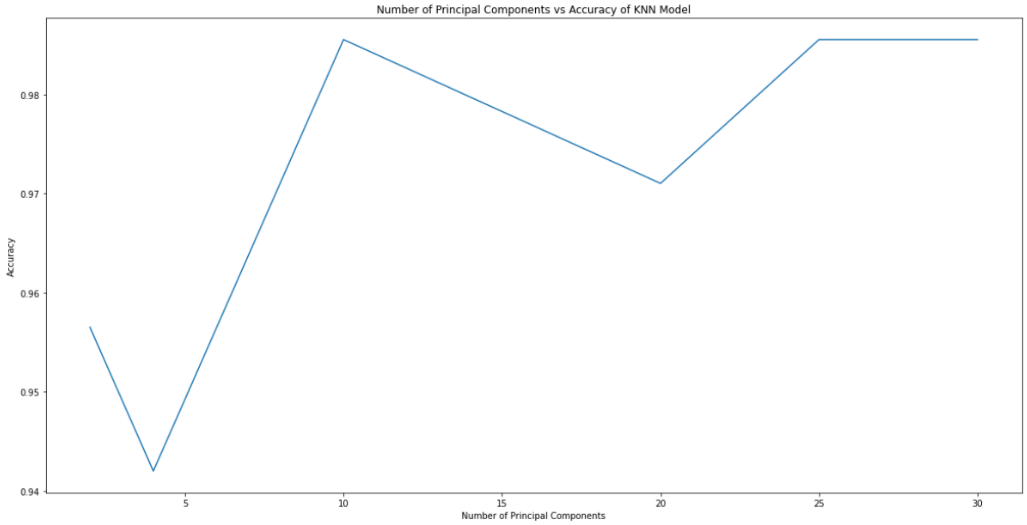
- **(5 points) Report the accuracy of the NEW testing dataset when using PCA (p = 10) with 5NN. To show your work, please submit the corresponding .csv file (including the name of .csv file in your answer below). Your .csv file should have 12 columns: columns 1-10 are the 10 principal components, column 11 is the original ground truth output “Class", and the last column is the predicted output “Class".**

  The .csv file titled ‘test_output_norm.csv’ can be found in the following Github repository under the H2 directory.

• (6 points) Plot your results by varying p: 2, 4, 8, 10, 20, 25, and 30 respectively. In your plot, the x-axis represents the number of principal components and the y-axis refers to the accuracy of the NEW testing dataset using the corresponding number of principal components and 5NN.



```
# Plotting Number of Principal Components vs Accuracy of the KNN Model
plt.figure(figsize=(20,10))
plt.plot(p_values, accuracy_list)
plt.ylabel("Accuracy")
plt.xlabel("Number of Principal Components")
plt.title("Number of Principal Components vs Accuracy of KNN Model")
plt.show()
```

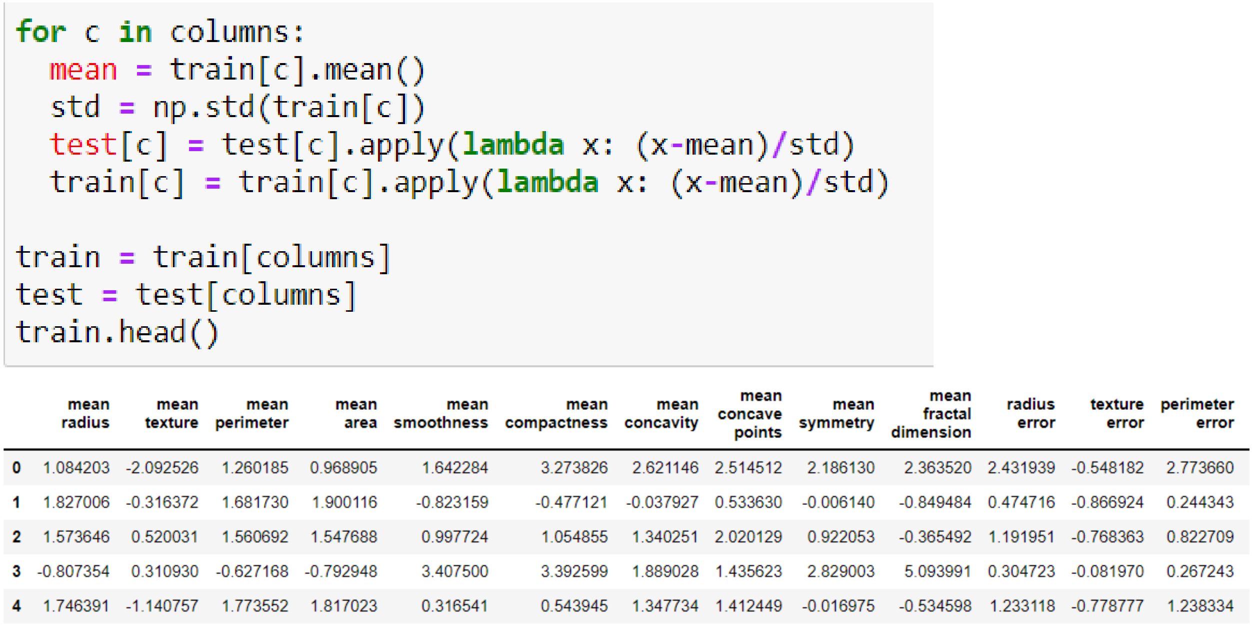


• (2 points) Based upon the (PCA + 5NN)'s results above, what is the most “reasonable" number of principal components among all the choices? Justify your answer.

Choosing 10 principal components is the ideal choice in this scenario since it gives us the highest accuracy per the number of components (n) used. After the number of principal components crosses 10, the accuracy decreases and regains shape at n = 25. But choosing 10 makes the most sense since the purpose of PCA is to trim the number of input features used when training the model whilst maximizing accuracy.

c) (18 points) Preprocess Data-Standardization: Similarly, please run standardization on all input features to obtain the standardized training and the standardized testing datasets. Then repeat the four steps i-iv in (b) above on the two NEW standardized datasets.

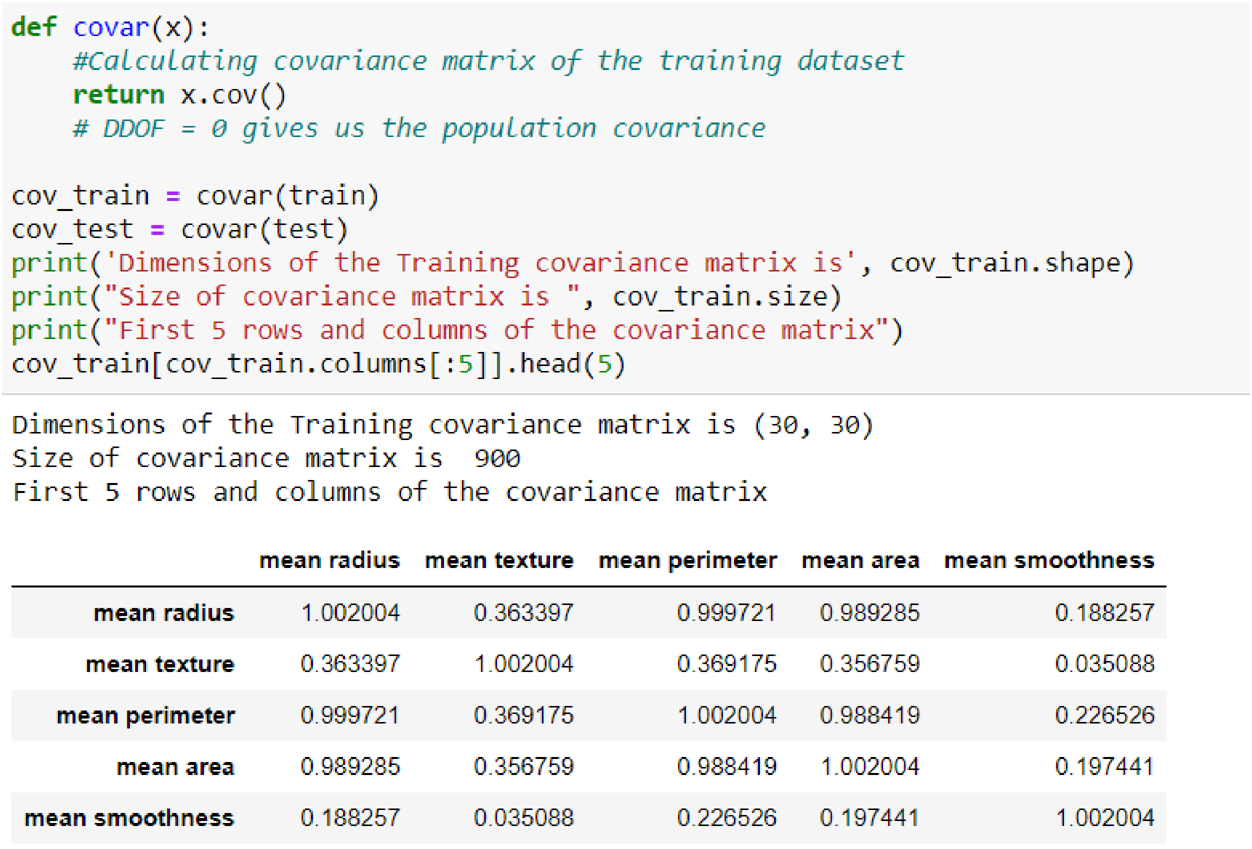
Standardized Training dataset



```
for c in columns:
  mean = train[c].mean()
  std = np.std(train[c])
  test[c] = test[c].apply(lambda x: (x-mean)/std)
  train[c] = train[c].apply(lambda x: (x-mean)/std)

train = train[columns]
test = test[columns]
train.head()
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness	mean compactness	mean concavity	mean concave points	mean symmetry	mean fractal dimension	radius error	texture error	perimeter error
0	1.084203	-2.092526	1.260185	0.968905	1.642284	3.273826	2.621146	2.514512	2.186130	2.363520	2.431939	-0.548182	2.773660
1	1.827006	-0.316372	1.681730	1.900116	-0.823159	-0.477121	-0.037927	0.533630	-0.006140	-0.849484	0.474716	-0.866924	0.244343
2	1.573646	0.520031	1.560692	1.547688	0.997724	1.054855	1.340251	2.020129	0.922053	-0.365492	1.191951	-0.768363	0.822709
3	-0.807354	0.310930	-0.627168	-0.792948	3.407500	3.392599	1.889028	1.435623	2.829003	5.093991	0.304723	-0.081970	0.267243
4	1.746391	-1.140757	1.773552	1.817023	0.316541	0.543945	1.347734	1.412449	-0.016975	-0.534598	1.233118	-0.778777	1.238334

i. (2 points) Calculate the covariance matrix of the NEW training dataset. Please 1) specify the dimension of the resulted covariance matrix and 2) given the space limitation, please report the first 5 * 5 of the covariance matrix, that is, only reporting the first five rows and the first five columns of the entire covariance Matrix.

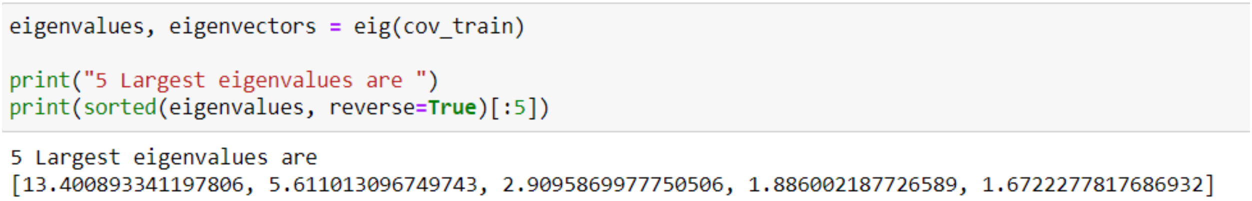


```
def covar(x):
    #Calculating covariance matrix of the training dataset
    return x.cov()
    # DDOF = 0 gives us the population covariance

cov_train = covar(train)
cov_test = covar(test)
print('Dimensions of the Training covariance matrix is', cov_train.shape)
print("Size of covariance matrix is ", cov_train.size)
print("First 5 rows and columns of the covariance matrix")
cov_train[cov_train.columns[:5]].head(5)
```

```
Dimensions of the Training covariance matrix is (30, 30)
Size of covariance matrix is  900
First 5 rows and columns of the covariance matrix
```

	mean radius	mean texture	mean perimeter	mean area	mean smoothness
mean radius	1.002004	0.363397	0.999721	0.989285	0.188257
mean texture	0.363397	1.002004	0.369175	0.356759	0.035088
mean perimeter	0.999721	0.369175	1.002004	0.988419	0.226526
mean area	0.989285	0.356759	0.988419	1.002004	0.197441
mean smoothness	0.188257	0.035088	0.226526	0.197441	1.002004

ii. (2 points) Calculate the eigenvalues and the eigenvectors based on the entire covariance matrix in (i) above. Report the size of the covariance matrix and the 5 largest eigenvalues.

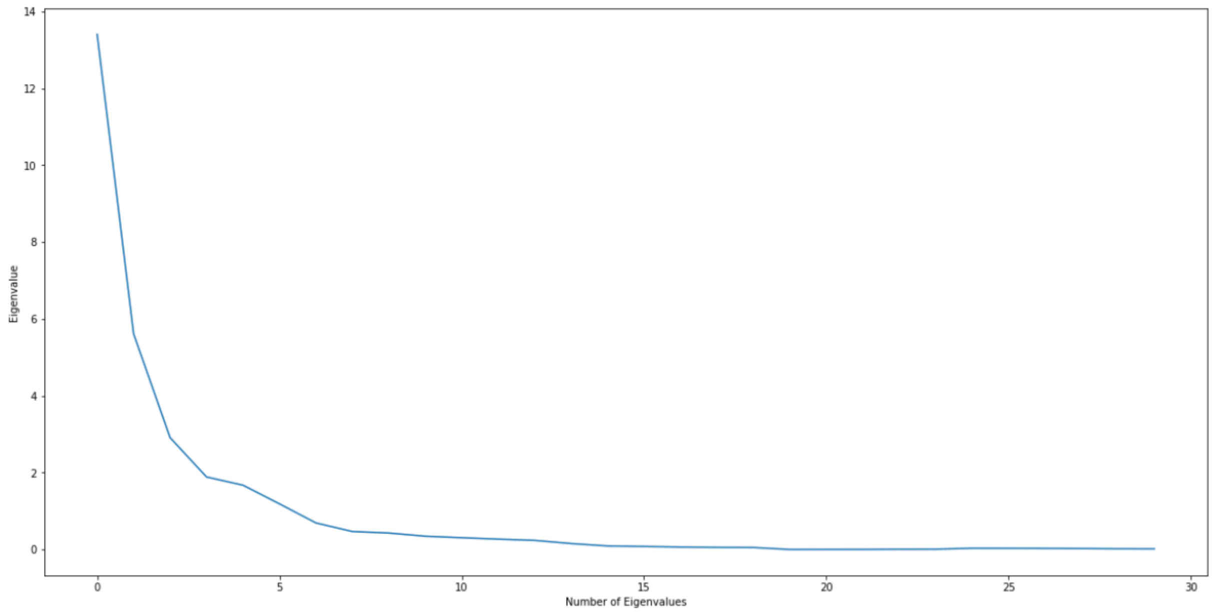


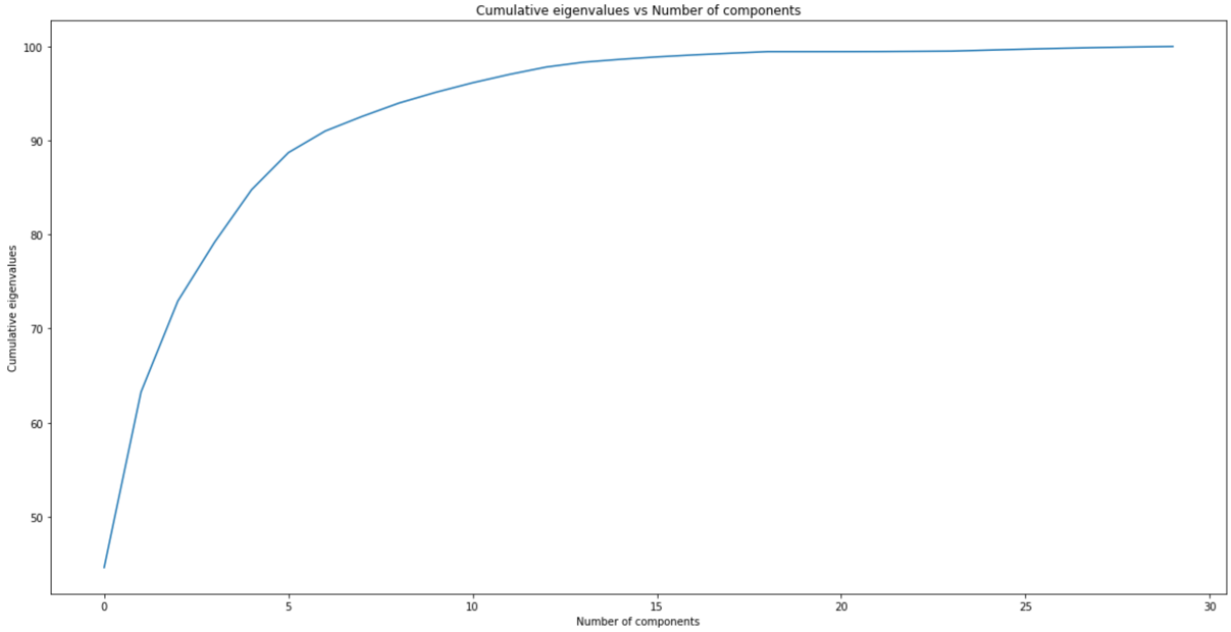
```
eigenvalues, eigenvectors = eig(cov_train)

print("5 Largest eigenvalues are ")
print(sorted(eigenvalues, reverse=True)[:5])
```

```
5 Largest eigenvalues are
[13.400893341197806, 5.611013096749743, 2.9095869977750506, 1.886002187726589, 1.6722277817686932]
```

iii. (1 point) Display the eigenvalues using a bar graph or a plot, and choose a reasonable number(s) of eigenvectors. Justify your answer.

Upon analyzing the scree plot, we see that the magnitude of the slope of the curve decreases significantly after the 7th eigenvector. After the 7th eigenvector, the curve straightens out implying that the first 7 eigenvectors encode most of the important information required for PCA. (Note: The plot starts with index = 0, so the 6th eigenvector is in fact the 7th in the graph).




Cumulative eigenvalues vs Number of components

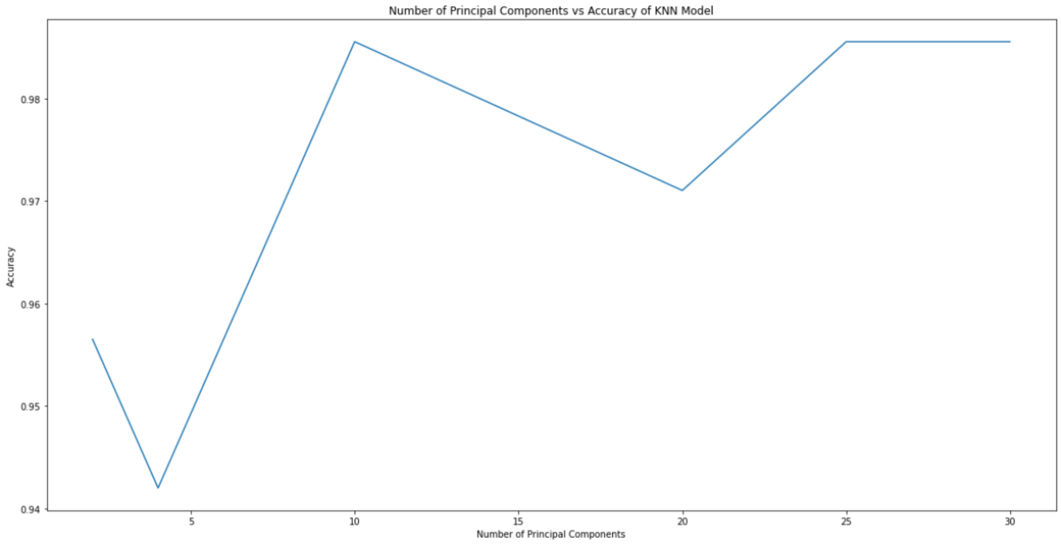
iv. (13 points) Next, you will combine PCA with a K-nearest neighbor (KNN) classifier. More specifically, PCA will be applied to reduce the dimensionality of data by transforming the original data into p (p 30) principal components; and then KNN (K = 5, Euclidean distance as distance metric) will be employed to the p principal components for classification.

- (5 points) Report the accuracy of the NEW testing dataset when using PCA (p = 10) with 5NN. To show your work, please submit the corresponding .csv file (including the name of .csv file in your answer below). Your .csv file should have 12 columns: columns 1-10 are the 10 principal

components, column 11 is the original ground truth output “Class", and the last column is the predicted output “Class".

The .csv file titled ‘test_output_std.csv’ can be found in the following Github repository under the H2 directory.

• (6 points) Plot your results by varying p: 2, 4, 8, 10, 20, 25, and 30 respectively. In your plot, the x-axis represents the number of principal components and the y-axis refers to the accuracy of the NEW testing dataset using the corresponding number of principal components and 5NN.



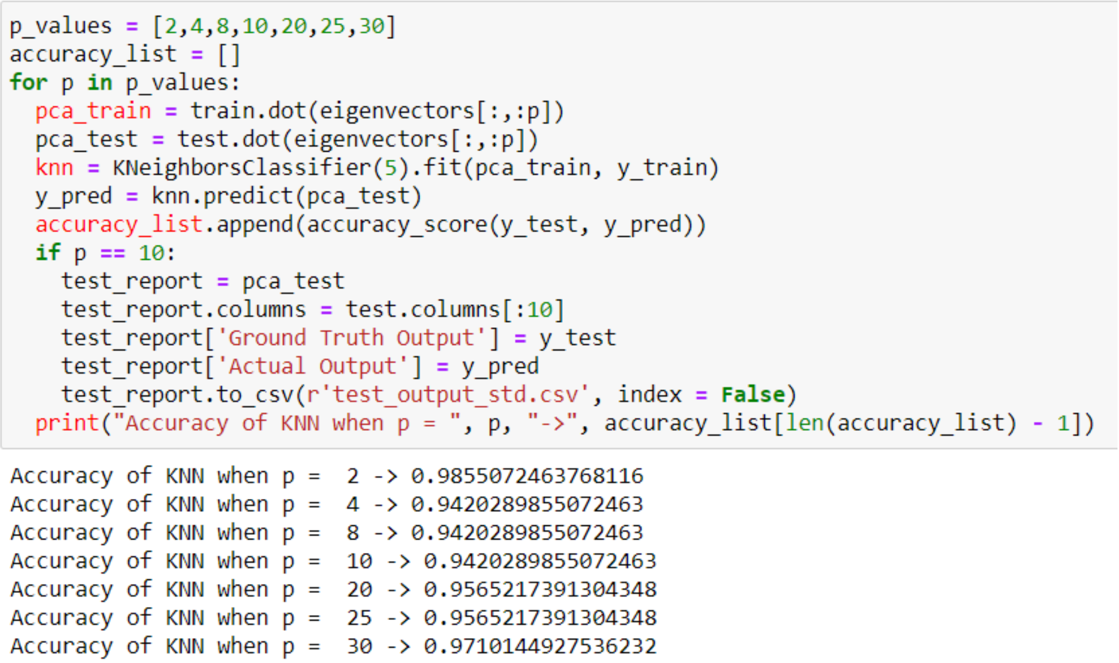
• (2 points) Based upon the (PCA + 5NN)'s results above, what is the most “reasonable" number of principal components among all the choices? Justify your answer.

Choosing 2 principal components is the ideal choice in this scenario since it gives us the highest accuracy per the number of components (n) used. After the number of principal components crosses 2, the accuracy decreases rapidly and increases steadily until n = 30 but is still considerably lesser than the accuracy that n = 2 offers. But choosing 10 makes the most sense since the purpose of PCA is to trim the number of input features used when training the model whilst maximizing accuracy.

(d) (2 points) Comparing the results from (b) and (c), which of the two data-processing procedures, normalization or standardization, would you prefer for the given datasets? And why? (Answer without any justification will get zero points.)

Here, we prefer standardization to normalization since the first 2 principal components offer an accuracy of 98.55% compared to normalization’s 10 principal components which offer the same accuracy score(98.55%). Since our intent is to minimize the number of input features whilst increasing the accuracy of the model, we choose Standardization over Normalization.

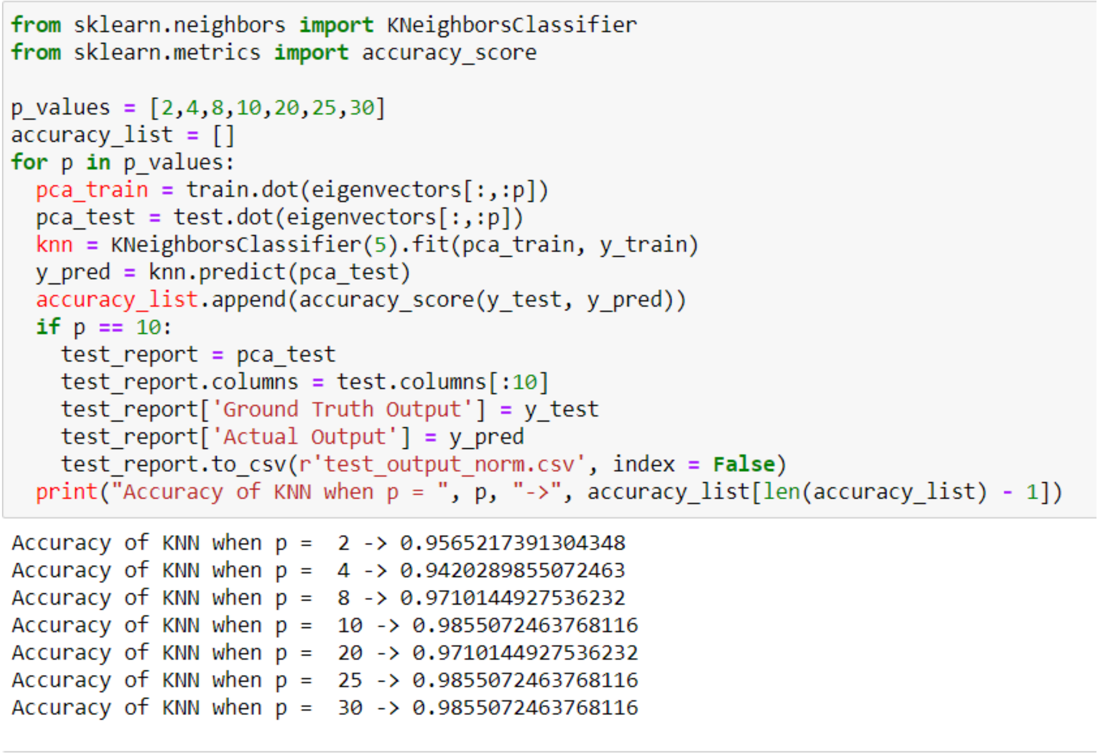
Accuracy scores for Standardized dataset



```
p_values = [2,4,8,10,20,25,30]
accuracy_list = []
for p in p_values:
  pca_train = train.dot(eigenvectors[:,:p])
  pca_test = test.dot(eigenvectors[:,:p])
  knn = KNeighborsClassifier(5).fit(pca_train, y_train)
  y_pred = knn.predict(pca_test)
  accuracy_list.append(accuracy_score(y_test, y_pred))
  if p == 10:
    test_report = pca_test
    test_report.columns = test.columns[:10]
    test_report['Ground Truth Output'] = y_test
    test_report['Actual Output'] = y_pred
    test_report.to_csv(r'test_output_std.csv', index = False)
  print("Accuracy of KNN when p = ", p, "->", accuracy_list[len(accuracy_list) - 1])
```

```
Accuracy of KNN when p =  2 -> 0.9855072463768116
Accuracy of KNN when p =  4 -> 0.9420289855072463
Accuracy of KNN when p =  8 -> 0.9420289855072463
Accuracy of KNN when p =  10 -> 0.9420289855072463
Accuracy of KNN when p =  20 -> 0.9565217391304348
Accuracy of KNN when p =  25 -> 0.9565217391304348
Accuracy of KNN when p =  30 -> 0.9710144927536232
```

Accuracy scores for Standardized dataset



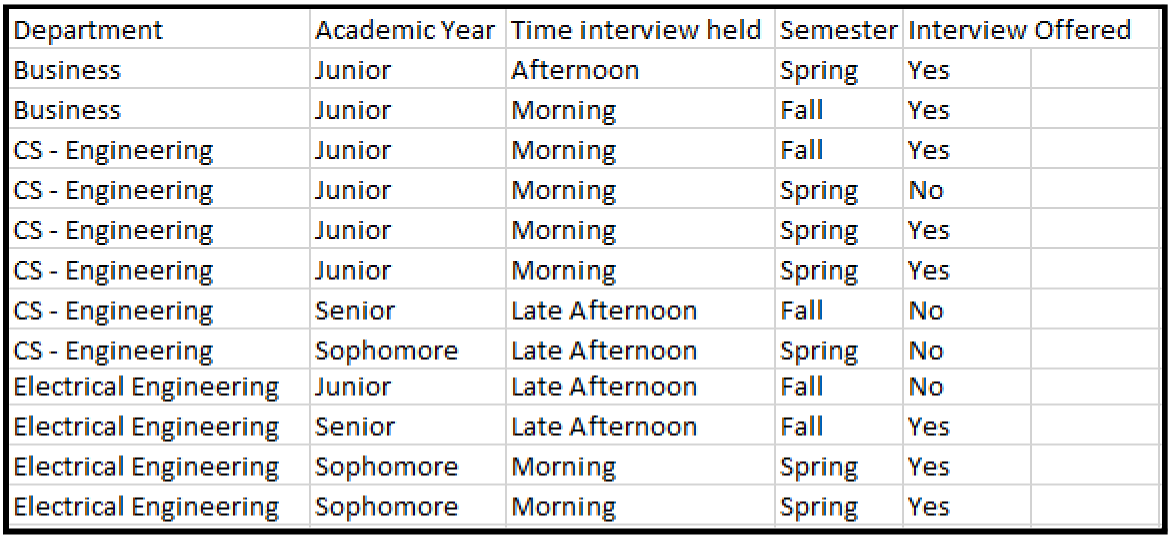
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

p_values = [2,4,8,10,20,25,30]
accuracy_list = []
for p in p_values:
  pca_train = train.dot(eigenvectors[:,:p])
  pca_test = test.dot(eigenvectors[:,:p])
  knn = KNeighborsClassifier(5).fit(pca_train, y_train)
  y_pred = knn.predict(pca_test)
  accuracy_list.append(accuracy_score(y_test, y_pred))
  if p == 10:
    test_report = pca_test
    test_report.columns = test.columns[:10]
    test_report['Ground Truth Output'] = y_test
    test_report['Actual Output'] = y_pred
    test_report.to_csv(r'test_output_norm.csv', index = False)
  print("Accuracy of KNN when p = ", p, "->", accuracy_list[len(accuracy_list) - 1])
```

```
Accuracy of KNN when p =  2 -> 0.9565217391304348
Accuracy of KNN when p =  4 -> 0.9420289855072463
Accuracy of KNN when p =  8 -> 0.9710144927536232
Accuracy of KNN when p =  10 -> 0.9855072463768116
Accuracy of KNN when p =  20 -> 0.9710144927536232
Accuracy of KNN when p =  25 -> 0.9855072463768116
Accuracy of KNN when p =  30 -> 0.9855072463768116
```

2. (30 points) [Decision Tree] [Angela Zhang] For this exercise, you will use Internships.csv, from 12 students who applied for internships.

Complete the following tasks using the decision tree algorithm discussed in the lecture.
Note that multiple-way splitting is allowed. In the case of ties, break ties in favor of the leftmost attribute. (You can hand-draw all of your trees on paper and scan your results into the nal PDF.)

(a) (15 points) Construct the tree manually using ID3/entropy computations, write down the computation process by showing the number of cases in each class for each node before splitting and show your tree step by step. (No partial credit)



Department	Academic Year	Time interview held	Semester	Interview Offered
Business	Junior	Afternoon	Spring	Yes
Business	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Spring	No
CS - Engineering	Junior	Morning	Spring	Yes
CS - Engineering	Junior	Morning	Spring	Yes
CS - Engineering	Senior	Late Afternoon	Fall	No
CS - Engineering	Sophomore	Late Afternoon	Spring	No
Electrical Engineering	Junior	Late Afternoon	Fall	No
Electrical Engineering	Senior	Late Afternoon	Fall	Yes
Electrical Engineering	Sophomore	Morning	Spring	Yes
Electrical Engineering	Sophomore	Morning	Spring	Yes

Level 0

Let Entropy be defined by "H".

$$H(X) = \sum - p(X = C)\, log\ _2 p\ (X = C)$$

$$H(Interview\, offered) = \ - P(I.O = Yes)\, log\ _2 (P(I.O = Yes)\ - P(I.O = No)\, log\ _2 (P(I.O = No))$$

$$H(Interview\, offered) = \ - (8/12)\, log\ _2 (8/12))\ - \ (4/12)\, log_2 (4/12))$$

0.39 + 0.528
0.918 = **0.92** (approx)

Department:

H(Interview Offered| Department = Business) = $H(X) = \sum - p(X = C)\, log\ _2 p\ (X = C)$

-P(Interview Offered = Yes | Department = Business) $log\ _2$ P(Interview Offered = Yes | Department = Business) -P(Interview Offered = No | Department = Business) $log\ _2$ P(Interview Offered = No | Department = Business)

- 1 log_{2} 1- 0

= 0

H(Interview|Department=CS-Engineering)= $H(X) = \sum - p(X = C)\, log_{2} p\,(X = C)$

-P(Interview Offered = Yes | Department = CS - Engineering) log_{2} P(Interview Offered = Yes | Department = CS - Engineering) -P(Interview Offered = No | Department = CS - Engineering) log_{2} P(Interview Offered = No | Department = CS - Engineering)

- 1/2 $log_{2} 1/2$- 1/2 log_{2} 1/2

= 1

H(Interview Offered | Department = Electrical Engineering)=
-P(Interview Offered = Yes | Department = Electrical Engineering) log_{2} P(Interview Offered = Yes |Department = Electrical Engineering) - P(Interview Offered = No | Department = Electrical Engineering) log_{2} P(Interview Offered = No | Department = Electrical Engineering)

- 3/4 $log_{2} 3/4$- 3/4 log_{2} 3/4

= 0.81

Information Gain (Department):

IG= H(Interview Offered) - P(Department = Business)H(Department = Business) - P(Department = CS-Engineering)H(Department = CS-Engineering) - P(Department= Electrical Engineering)H(Department = Electrical Engineering)

P(Department = Business)=2/12
P(Department = CS-Engineering)=6/12
P(Department = Electrical Engineering)=4/12

0.92- 2/12(0) - 6/12(1) - 4/12(0.81)
= 0.15027
≃ **0.15**

Academic Year

H(Academic Year = Junior)= $H(X) = \sum - p(X = C)\, log_{2} p\,(X = C)$

-P(Interview Offered = Yes | Academic Year= Junior) log_{2} P(Interview Offered = Yes | Academic Year= Junior) -P(Interview Offered = No | Academic Year= Junior) log_{2} P(Interview Offered = No | Academic Year= Junior)

- 5/7 log_{2} 5/7-2/7log_{2} 2/7 **=** 0.346+0.516=0.862

≃ **0.87**

H(Academic Year= Senior)= $H(X) = \sum - p(X = C)\, log_{2} p\,(X = C)$

-P(Interview Offered = Yes | Academic Year= Senior) $log\ _2$P(Interview Offered = Yes | Academic Year= Senior) --P(Interview Offered = No | Academic Year= Senior) $log\ _2$ P(Interview Offered = No | Academic Year= Senior)
- 1/2 $log\ _2 1/2$- 1/2 $log\ _2$ 1/2
= 1

H(Academic Year= Sophomore)= $H(X)\ = \sum -\ p(X\ =\ C)\ log\ _2 p\ (X\ =\ C)$

-P(Interview Offered = Yes | Academic Year= Sophomore) $log\ _2$P(Interview Offered = Yes | Academic Year= Sophomore) - P(Interview Offered = No | Academic Year= Sophomore) $log\ _2$P(Interview Offered = No | Academic Year= Sophomore)
- 1/3 $log\ _2 1/3$- 2/3 $log\ _2$2/3
= 0.92

<u>Information Gain (IG) Academic Year:</u>

IG= H(Interview Offered) - P(Academic Year= Junior)H(Academic Year= Junior) - P(Academic Year= Senior)H(Academic Year= Senior) - P(Academic Year= Sophomore)H(Academic Year= Sophomore)

P(Academic Year = Junior**)=7/12**
P(Academic Year = Senior**)=2/12**
P(Academic Year = Sophomore**)=3/12**

IG(Academic Year) = 0.92- 7/12(0.87) - 2/12(1) - 3/12(0.92)
= **0.016**

<u>Time Interview Held (TI):</u>

Let TI denote Time Interview Held

H(TI=Afternoon)= $H(X)\ = \sum -\ p(X\ =\ C)\ log\ _2 p\ (X\ =\ C)$ =

-P(Interview Offered = Yes | TI=Afternoon) $log\ _2$P(Interview Offered = Yes | TI=Afternoon) -P(Interview Offered = No | TI=Afternoon) $log\ _2$P(Interview Offered = No | TI=Afternoon)
- 1 $log\ _2$1-0 $log\ _2$0
= 0

H(TI=Morning)= $H(X)\ = \sum -\ p(X\ =\ C)\ log\ _2 p\ (X\ =\ C)$ =

-P(Interview Offered = Yes | TI=Morning) $log\ _2$P(Interview Offered = Yes | TI=Morning) -P(Interview Offered = No | TI=Morning) $log\ _2$P(Interview Offered = No | TI=Morning)

- 6/7 $log_{2} 6/7$- 1/7 log_{2} 1/7
= 0.592

Let Late AN denote Late Afternoon

H(TI=Late AN)= $H(X) = \sum - p(X = C) \, log_{2} p\,(X = C)$

-P(Interview Offered = Yes | TI=Late AN) log_{2} P(Interview Offered = Yes | TI= Late AN) - P(Interview Offered = No | TI= Late AN) log_{2} P(Interview Offered = No | TI= Late AN)
- 1/4 $log_{2} 1/4$- 3/4 log_{2} 3/4
= 0.81

Information Gain (TI)

IG= H(Interview Offered) - P(TI= Afternoon)H(TI= Afternoon) - P(TI= Morning)H(TI= Morning) - P(TI= Late AN)H(TI= Late AN)

P(TI=Afternoon**)=1/12**
P(TI=Morning**)=7/12**
P(TI=Late AN**)=4/12**

0.92- 1/12(0) - 7/12(0.592) - 4/12(0.81)
= **0.305**

Semester

H(Semester=Spring) = $H(X) = \sum - p(X = C) \, log_{2} p\,(X = C)$

-P(Interview Offered = Yes | Semester=Spring) log_{2} P(Interview Offered = Yes | Semester=Spring) -P(Interview Offered = No | Semester=Spring) log_{2} P(Interview Offered = No | Semester=Spring)
- 5/7 $log_{2} 5/7$- 2/7 log_{2} 2/7
= 0.87

H(Semester=Fall) = $H(X) = \sum - p(X = C) \, log_{2} p\,(X = C)$

-P(Interview Offered = Yes | Semester=Fall) log_{2} P(Interview Offered = Yes | Semester=Fall) -P(Interview Offered = No | Semester=Fall) log_{2} P(Interview Offered = No | Semester=Fall)
- 3/5 $log_{2} 3/5$- 2/5 log_{2} 2/5
= 0.97

Information Gain (Semester):

IG= H(Interview Offered) - P(Semester = Spring)H(Semester = Spring) - P(Semester = Fall)H(Semester = Fall)

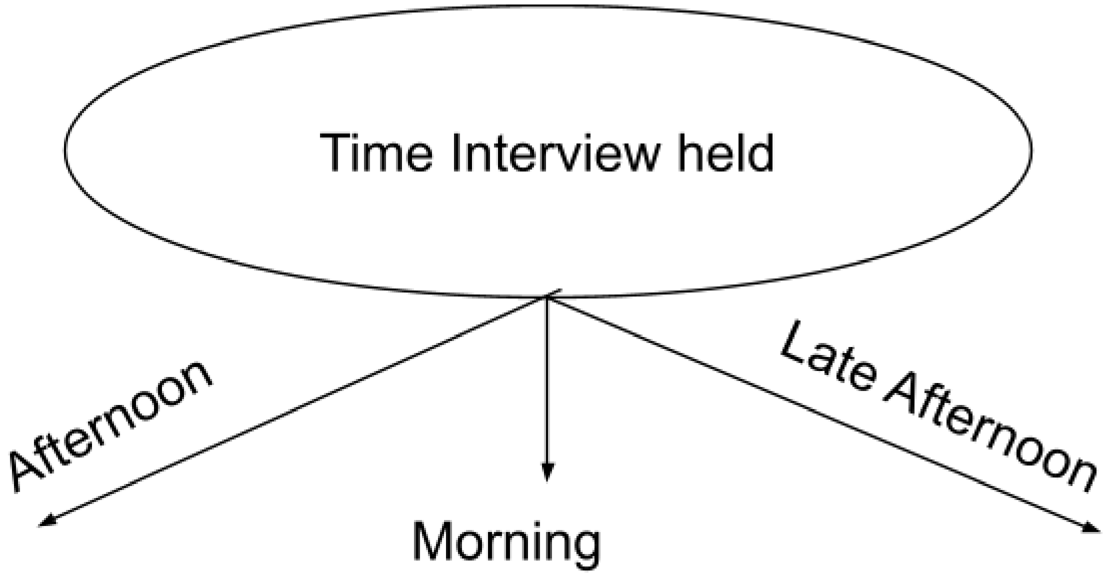
P(Semester=Spring**)=7/12**

P(Semester=Fall**)=5/12**

0.92- (7/12)*(0.87) - (5/12)*(0.97) = **0.008**

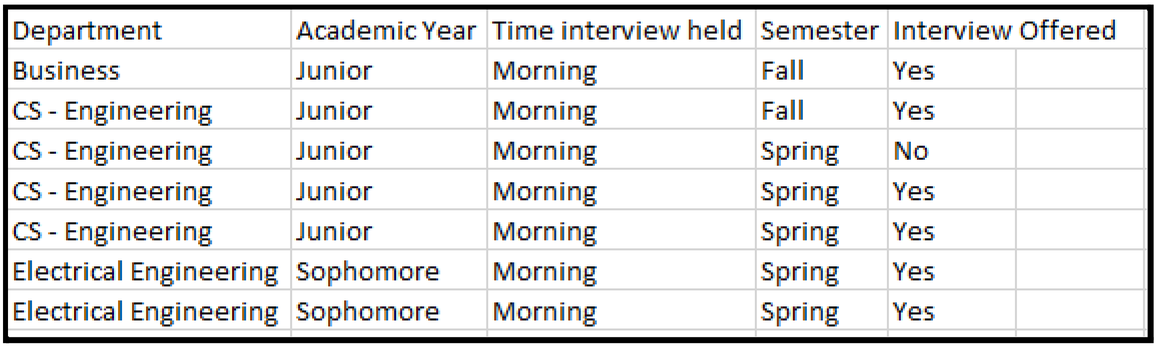
Attribute	**Information Gain**
Department	**0.15**
Academic	**0.016**
Time Interview Held (*)	**0.305 (*)**
Semester	**0.008**

We select the attribute with the highest information gain. In our case, we select Time Interview Held.



LEVEL 1

Time Interview held = Morning



Department	Academic Year	Time interview held	Semester	Interview Offered
Business	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Spring	No
CS - Engineering	Junior	Morning	Spring	Yes
CS - Engineering	Junior	Morning	Spring	Yes
Electrical Engineering	Sophomore	Morning	Spring	Yes
Electrical Engineering	Sophomore	Morning	Spring	Yes

$H(Interview\ offered\,|\,TI\ =\ Morning)\ =$
$-\ P(Yes)\ log\ _2(P(Yes)\ -\ P(No)\ log\ _2(P(No))$
$-\ (6/7)\ log\ _2(6/7)\ -\ (1/7)\ log_2(1/7)$
= **0.59**

Department

H(Department = Business) =
-P(Interview Offered = Yes | Department = Business) $log\ _2$ P(Interview Offered = Yes | Department = Business) -P(Interview Offered = No | Department = Business) $log\ _2$ P(Interview Offered = No | Department = Business)
$-\ (1)\ log\ _2 1\ -\ 0$
= 0

H(Department = CS - Engineering) =
-P(Interview Offered = Yes | Department = CS - Engineering) $log\ _2$ P(Interview Offered = Yes | Department = CS - Engineering) -P(Interview Offered = No | Department = CS - Engineering) $log\ _2$ P(Interview Offered = No | Department = CS - Engineering)

$-\ (3/4)\ log\ _2(3/4)\ -\ (1/4)\ log_2(1/4)$
=**0.81**

H(Department = Electrical) =
-P(Interview Offered = Yes | Department = Electrical Engineering) $log\ _2$ P(Interview Offered = Yes |Department = Electrical Engineering) - P(Interview Offered = No | Department =

Electrical Engineering) log_2 P(Interview Offered = No | Department = Electrical Engineering)

$-(2/2)\ log_2(2/2) - 0$

= 0

Information Gain (Department):

IG = H(Interview Offered | TI = Morning) - P(Department = Business)H(Department = Business) - P(Department = CS-Engineering)H(Department = CS-Engineering) - P(Department= Electrical Engineering)H(Department = Electrical Engineering)

0.592- (1/7)(0) - (4/7)(0.81) - (2/7)(0)
=0.129

<u>Semester</u>

H(Semester=Spring)

-P(Interview Offered = Yes | Semester=Spring) log_2 P(Interview Offered = Yes | Semester=Spring) -P(Interview Offered = No | Semester=Spring) log_2 P(Interview Offered = No | Semester=Spring)
$-(4/5)\ log_2(4/5)\ -\ (1/5)\ log_2(1/5)$
=**0.722**

H(Semester=Fall)

-P(Interview Offered = Yes | Semester=Fall) log_2 P(Interview Offered = Yes | Semester=Fall) -P(Interview Offered = No | Semester=Fall) log_2 P(Interview Offered = No | Semester=Fall)
$-(2/2)\ log_2(2/2) - 0$
= 0

Information Gain(Semester):
IG= H(Interview Offered | TI = Morning) - P(Semester = Spring)H(Semester = Spring) - P(Semester = Fall)H(Semester = Fall)

0.592 - (5/7)(0.722) - (2/7)(0)

=0.076

Academic Year

H(Academic Year = Sophomore) =

-P(Interview Offered = Yes | Academic Year= Sophomore) log_{2} P(Interview Offered = Yes | Academic Year= Sophomore) - P(Interview Offered = No | Academic Year= Sophomore) log_{2} P(Interview Offered = No | Academic Year= Sophomore)

$-(2/2)\,log_{2}(2/2) - 0$

= **0**

H(Academic Year = Junior) =

-P(Interview Offered = Yes | Academic Year= Junior) log_{2} P(Interview Offered = Yes | Academic Year= Junior) -P(Interview Offered = No | Academic Year= Junior) log_{2} P(Interview Offered = No | Academic Year= Junior)

$-(4/5)\,log_{2}(4/5) - (1/5)\,log_{2}(1/5)$ =**0.722**

Information Gain(Academic Year):

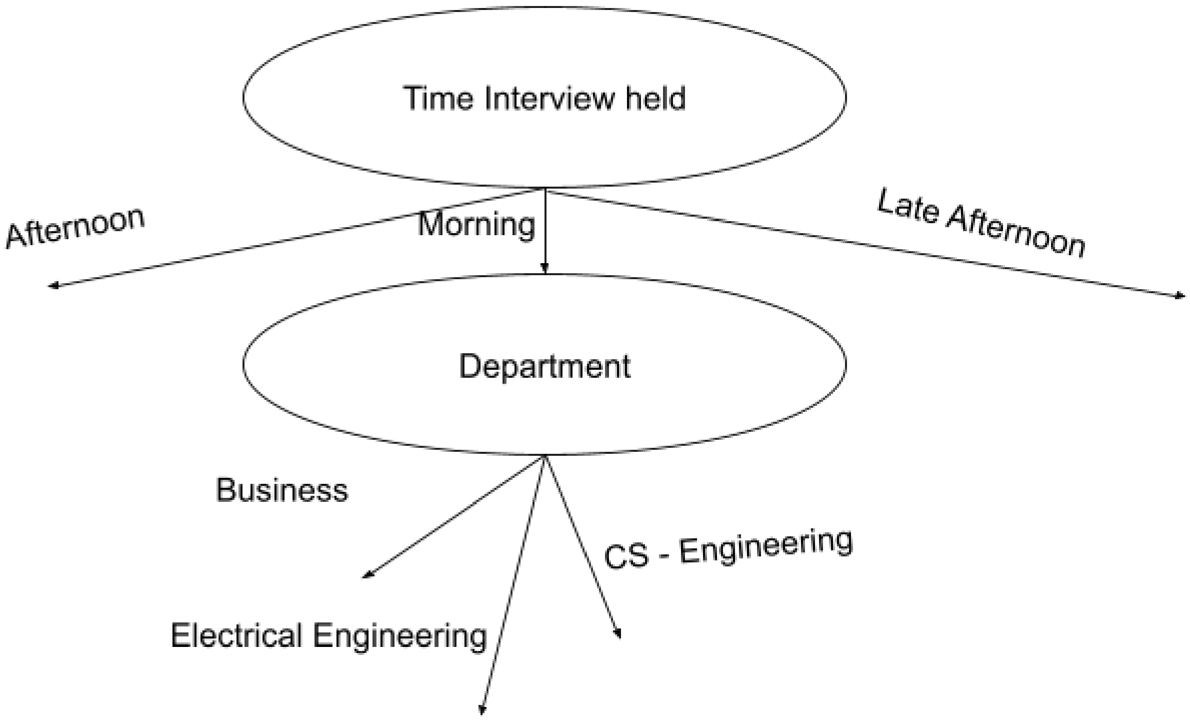
IG= H(Interview Offered | TI = Morning) - P(Academic Year= Junior)H(Academic Year= Junior) - P(Academic Year= Sophomore)H(Academic Year= Sophomore)

0.592 - (5/7)(0.722) - (2/7)(0)
=0.076

Attribute	**Information Gain**
Department (*)	**0.129 (*)**
Academic Year	**0.076**
Semester	**0.076**

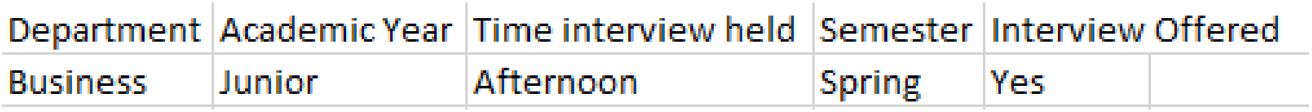
Department attribute has the higher Information Gain and is selected as the next node.

Root for Morning is Department (Higher Information Gain).

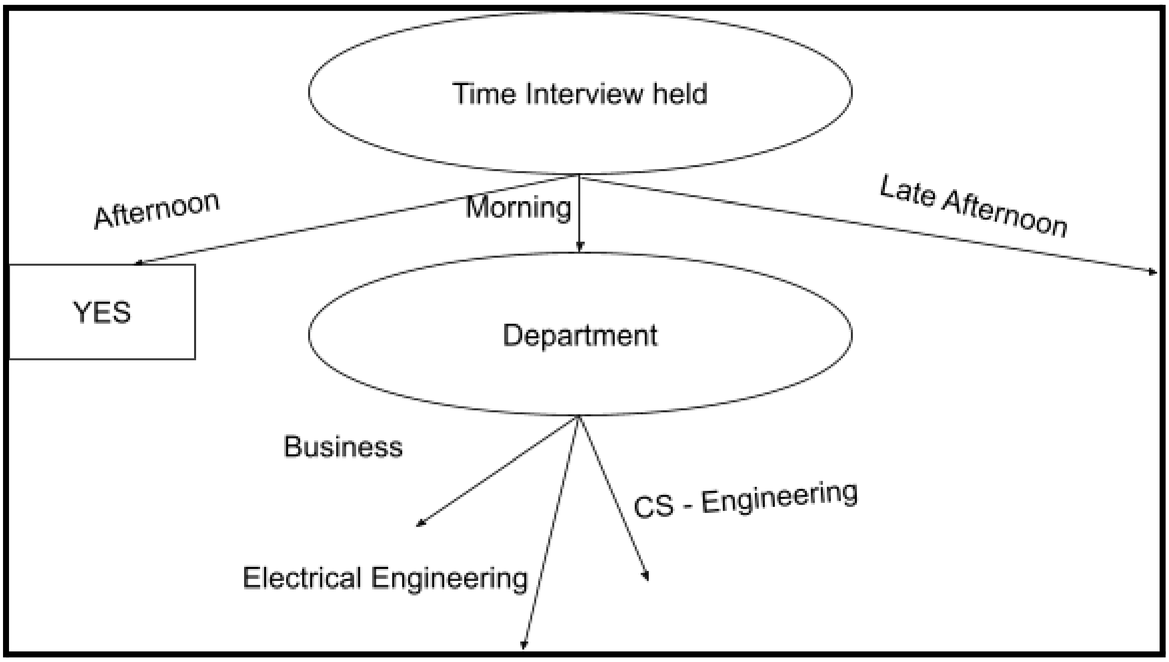


Time Interview Held = Afternoon

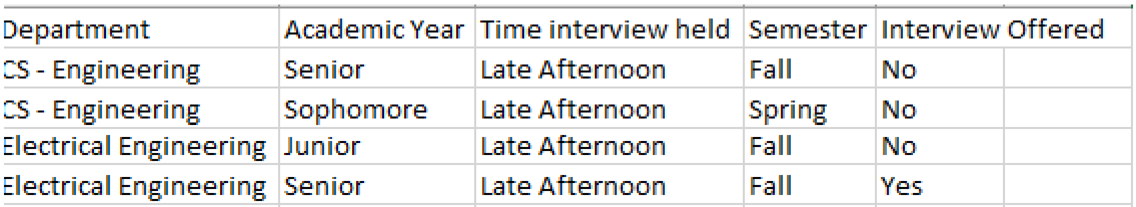
The attribute Afternoon has only one label, that is Yes. We therefore depict the leaf node for Afternoon as YES



Department	Academic Year	Time interview held	Semester	Interview Offered
Business	Junior	Afternoon	Spring	Yes



When Time Interview Held = Late Afternoon



Department	Academic Year	Time interview held	Semester	Interview Offered
CS - Engineering	Senior	Late Afternoon	Fall	No
CS - Engineering	Sophomore	Late Afternoon	Spring	No
Electrical Engineering	Junior	Late Afternoon	Fall	No
Electrical Engineering	Senior	Late Afternoon	Fall	Yes

$H(Interview\ offered\,|\,TI\ =\ Late\ Afternoon)\ =$
$-\ P(Yes)\ log\ _2(P(Yes)\ \ -\ P(No)\ log\ _2(P(No))$

$-\ (1/4)\ log\ _2(1/4)\ \ -\ \ (3/4)\ log_2(3/4)$

=**0.81**

Department:

H(Department = CS - Engineering) =

-P(Interview Offered = Yes | Department = CS - Engineering) $log\ _2$P(Interview Offered = Yes | Department = CS - Engineering) -P(Interview Offered = No | Department = CS - Engineering) $log\ _2$P(Interview Offered = No | Department = CS - Engineering)

$-\ (0)\ \ -\ \ (2/2)\ log_2(2/2)$

= 0

H(Department = EE)

-P(Interview Offered = Yes | Department = Electrical Engineering) log_{2} P(Interview Offered = Yes | Department = Electrical Engineering) -P(Interview Offered = No | Department = Electrical Engineering) log_{2} P(Interview Offered = No | Department = Electrical Engineering)

$-(1/2)\, log_{2}(1/2) - (1/2)\, log_{2}(1/2)$

= 1

Information Gain (IG Department)

IG = H(Interview Offered | TI = Late Afternoon) - P(Department = CS-Engineering)H(Department = CS-Engineering) - P(Department= Electrical Engineering)H(Department = Electrical Engineering)

0.81- (2/4)(0) - (2/4)(1)
0.81-0.5
=0.31

Academic Year

H(Academic Year = Senior) =

-P(Interview Offered = Yes | Academic Year= Senior) log_{2} P(Interview Offered = Yes | Academic Year= Senior) - P(Interview Offered = No | Academic Year= Senior) log_{2} P(Interview Offered = No | Academic Year= Senior)

$-(1/2)\, log_{2}(1/2) - (1/2)\, log_{2}(1/2)$

= **1**

H(Academic Year = Junior) =

-P(Interview Offered = Yes | Academic Year= Junior) log_{2} P(Interview Offered = Yes | Academic Year= Junior) -P(Interview Offered = No | Academic Year= Junior) log_{2} P(Interview Offered = No | Academic Year= Junior)

$-0 - (1) log_{2}(1)$

=**0**

H(Academic Year = Sophomore) =

-P(Interview Offered = Yes | Academic Year= Sophomore) log_2 P(Interview Offered = Yes | Academic Year= Sophomore) - P(Interview Offered = No | Academic Year= Sophomore) log_2 P(Interview Offered = No | Academic Year= Sophomore)

$- 0 - (1)log_2(1)$
= **0**

Information Gain(Academic Year):

IG= H(Interview Offered | TI = Late Afternoon) - P(Academic Year= Senior)H(Academic Year= Senior) - P(Academic Year= Junior)H(Academic Year= Junior) - P(Academic Year= Sophomore)H(Academic Year= Sophomore)

0.81 - (1/4)(0) - (2/4)(1) - (1/4)(0)
0.81-0.5
=0.31

Semester

H(Semester = Fall) =

-P(Interview Offered = Yes | Semester = Fall) log_2 P(Interview Offered = Yes | Semester = Fall) - P(Interview Offered = No | Semester = Fall) log_2 P(Interview Offered = No | Semester = Fall)

$- (1/3)\ log_2(1/3) - (2/3)\ log_2(2/3)$
=**0.92**

H(Semester = Spring) =

-P(Interview Offered = Yes | Semester = Spring) log_2 P(Interview Offered = Yes | Semester = Spring) - P(Interview Offered = No | Semester = Spring) log_2 P(Interview Offered = No | Semester = Spring)

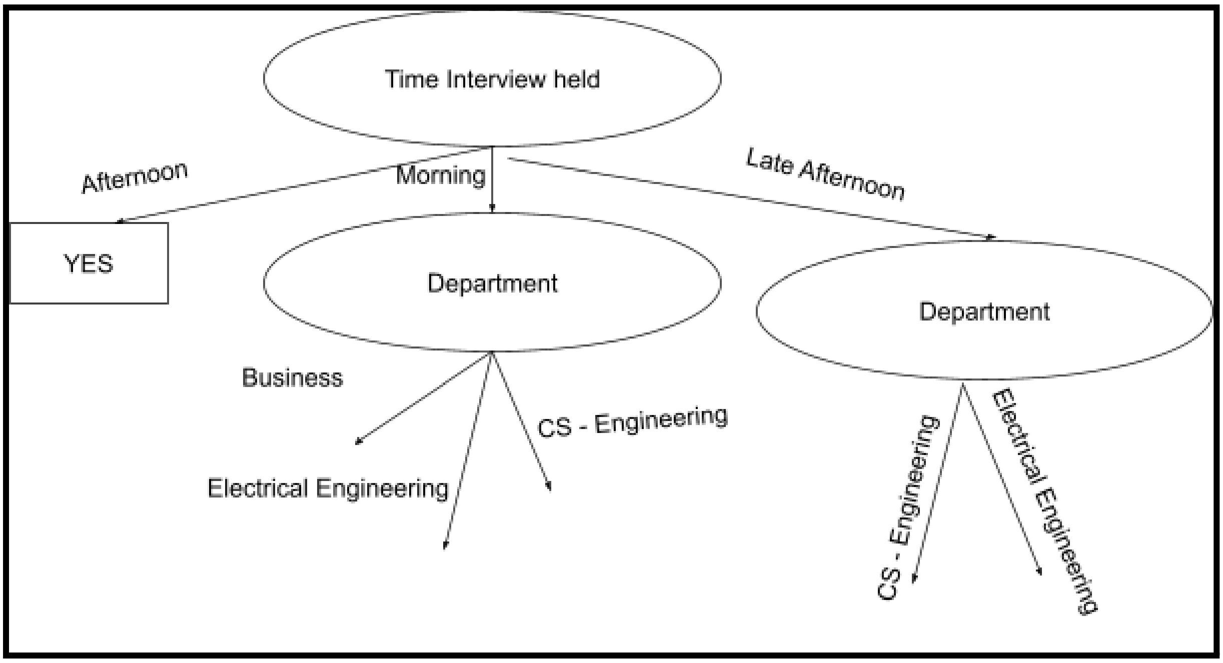
$- (0) - (1)\ log_2(1)$
= **0**

Information Gain (IG Semester)

IG= H(Interview Offered | TI = Late Afternoon) - P(Semester = Spring)H(Semester = Spring) - P(Semester = Fall)H(Semester = Fall)
=0.81 - (1/4)(0) - (3/4)(0.92)
=0.12

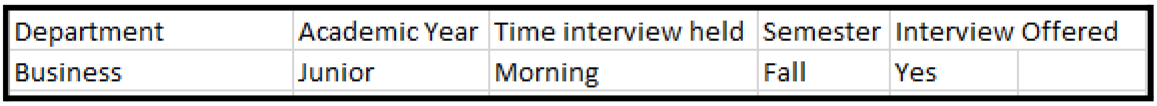
Attribute	**Information Gain**
Department (*)	**0.31 (*)**
Academic Year	**0.31**
Semester	**0.12**

<u>Next Root is Department, as it is the left most attribute and the highest information gain</u>



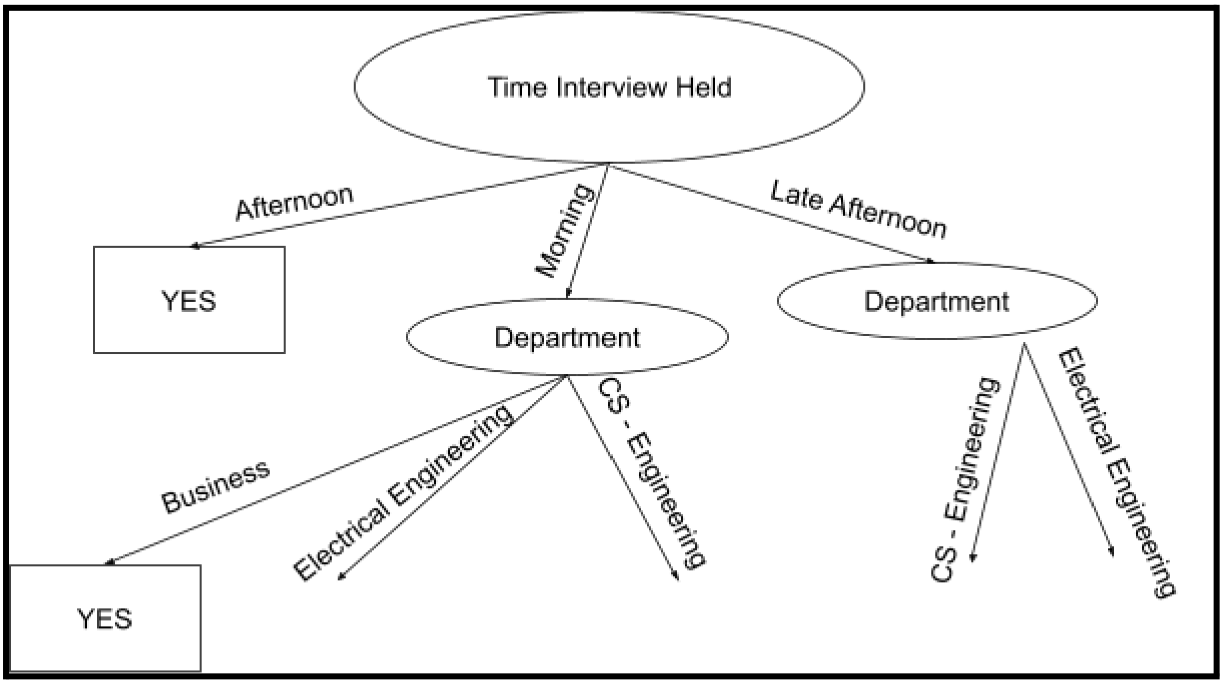
<u>LEVEL 2</u>

<u>When Time Interview Held = Morning and Department = Business</u>

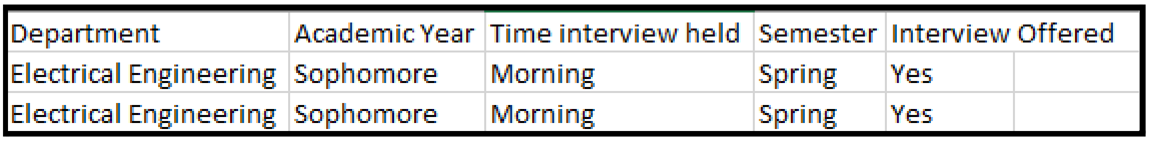


Department	Academic Year	Time interview held	Semester	Interview Offered
Business	Junior	Morning	Fall	Yes

When Time Interview Held = Morning , Department = Business , we have Interview offered = Yes. Therefore we set our leaf to YES

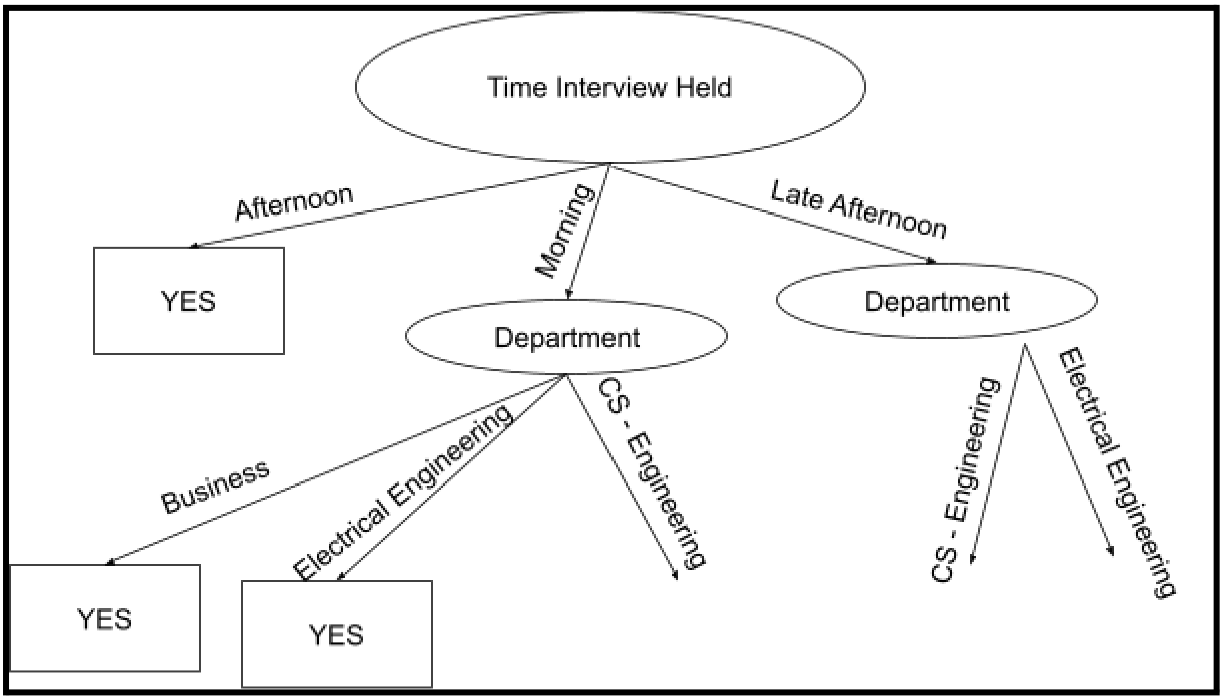


<u>When Time Interview Held = Morning and Department = Electrical Engineering</u>

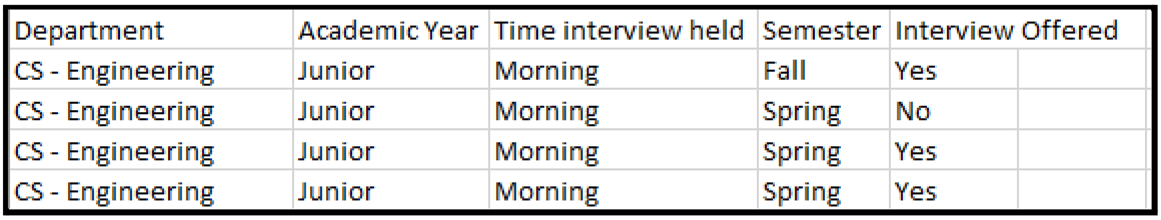


Department	Academic Year	Time interview held	Semester	Interview Offered	
Electrical Engineering	Sophomore	Morning	Spring	Yes	
Electrical Engineering	Sophomore	Morning	Spring	Yes	

When Time Interview Held = Morning , Department = Electrical Engineering , we have Interview offered = Yes. Therefore we set our leaf nodes to YES



When Time Interview Held = Morning and Department = CS - Engineering



Department	Academic Year	Time interview held	Semester	Interview Offered
CS - Engineering	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Spring	No
CS - Engineering	Junior	Morning	Spring	Yes
CS - Engineering	Junior	Morning	Spring	Yes

H(Interview Offered | TI = Morning and Department = CS- Engineering)= $-(3/4)\log_2(3/4) - (1/4)\log_2(1/4)$

=**0.81**

Academic Year:

H(Academic Year = Junior) =

-P(Interview Offered = Yes | Academic Year= Junior) $\log_2$ P(Interview Offered = Yes | Academic Year= Junior) -P(Interview Offered = No | Academic Year= Junior) $\log_2$ P(Interview Offered = No | Academic Year= Junior)

$-(3/4)\log_2(3/4) - (1/4)\log_2(1/4)$

=**0.81**

Information Gain (IG) =

H(Interview offered | TI = Morning and Department = CS Engineering)
- P(Academic Year = Junior) H (Academic Year = Junior) =
=0.81 - (4/4)(0.81)
0.81-0.81
=0

Semester

H(Semester = Fall)=
-P(Interview Offered = Yes | Semester = Fall) $log\ _2$ P(Interview Offered = Yes | Semester = Fall) - P(Interview Offered = No | Semester = Fall) $log\ _2$ P(Interview Offered = No | Semester = Fall)
$-\ (1)\ log\ _2(1)\ -\ 0$
= 0

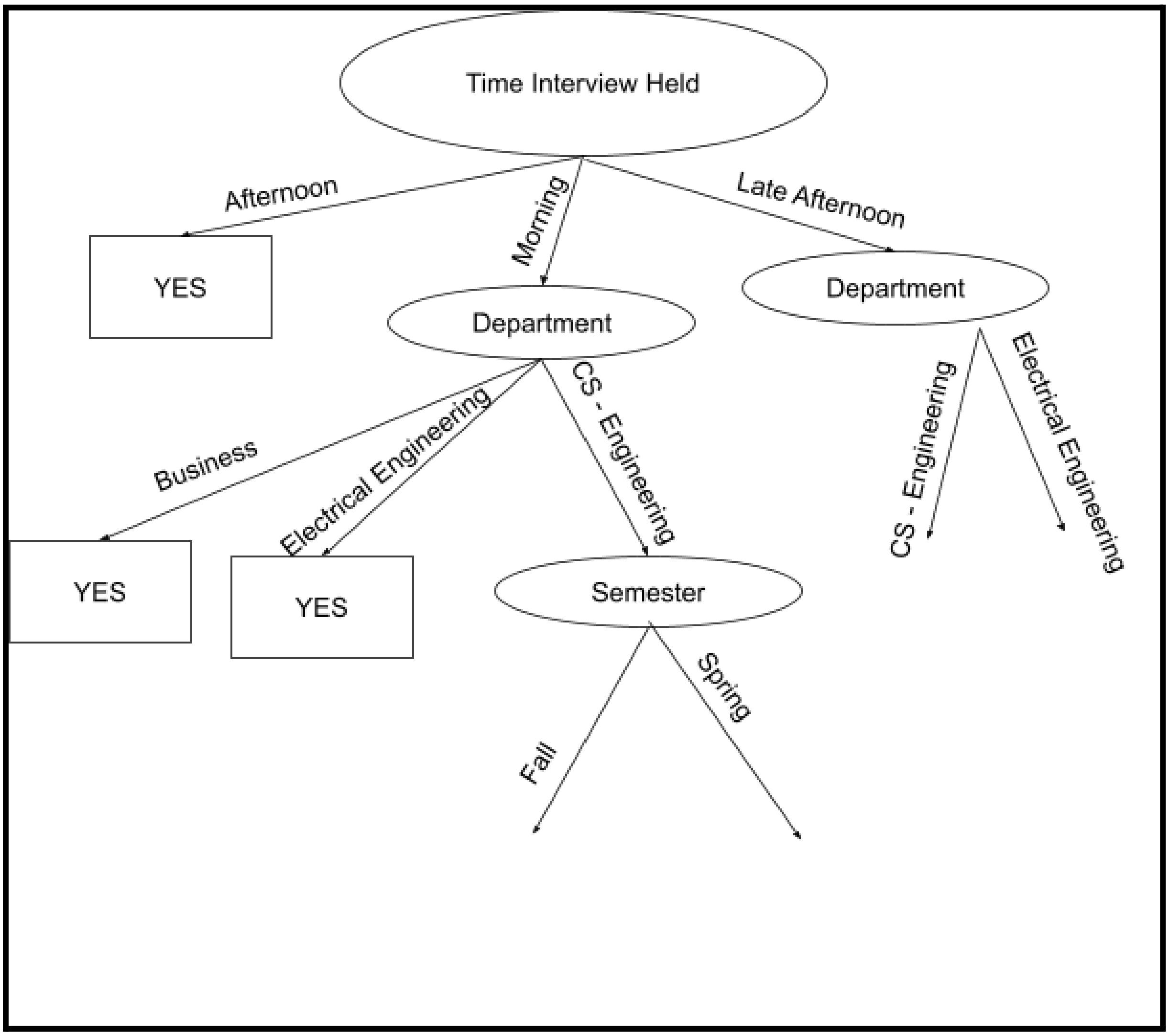
H(Semester = Spring)=
-P(Interview Offered = Yes | Semester = Spring) $log\ _2$ P(Interview Offered = Yes | Semester = Spring) - P(Interview Offered = No | Semester = Spring) $log\ _2$ P(Interview Offered = No | Semester = Spring) $-\ (2/3)\ log\ _2(2/3)\ -\ (1/3)\ log_2(1/3)$
= **0.92**

Information Gain (IG)=
IG= H(Interview Offered | TI = Morning and Dept = CS Engineering) - P(Semester = Spring)H(Semester = Spring) - P(Semester = Fall)H(Semester = Fall)

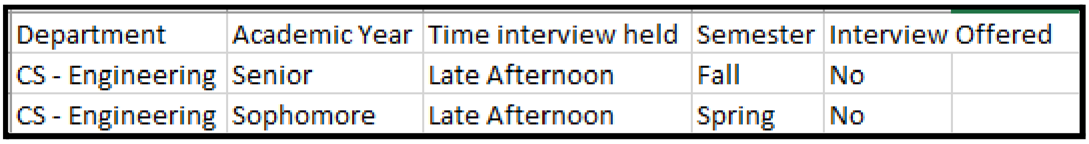
0.81 -(1/4)(0) - (3/4)(0.92) - (1/4)(0)
= 0.12

Attribute	**Information Gain**
Academic Year	**0**
Semester (*)	**0.12 (*)**

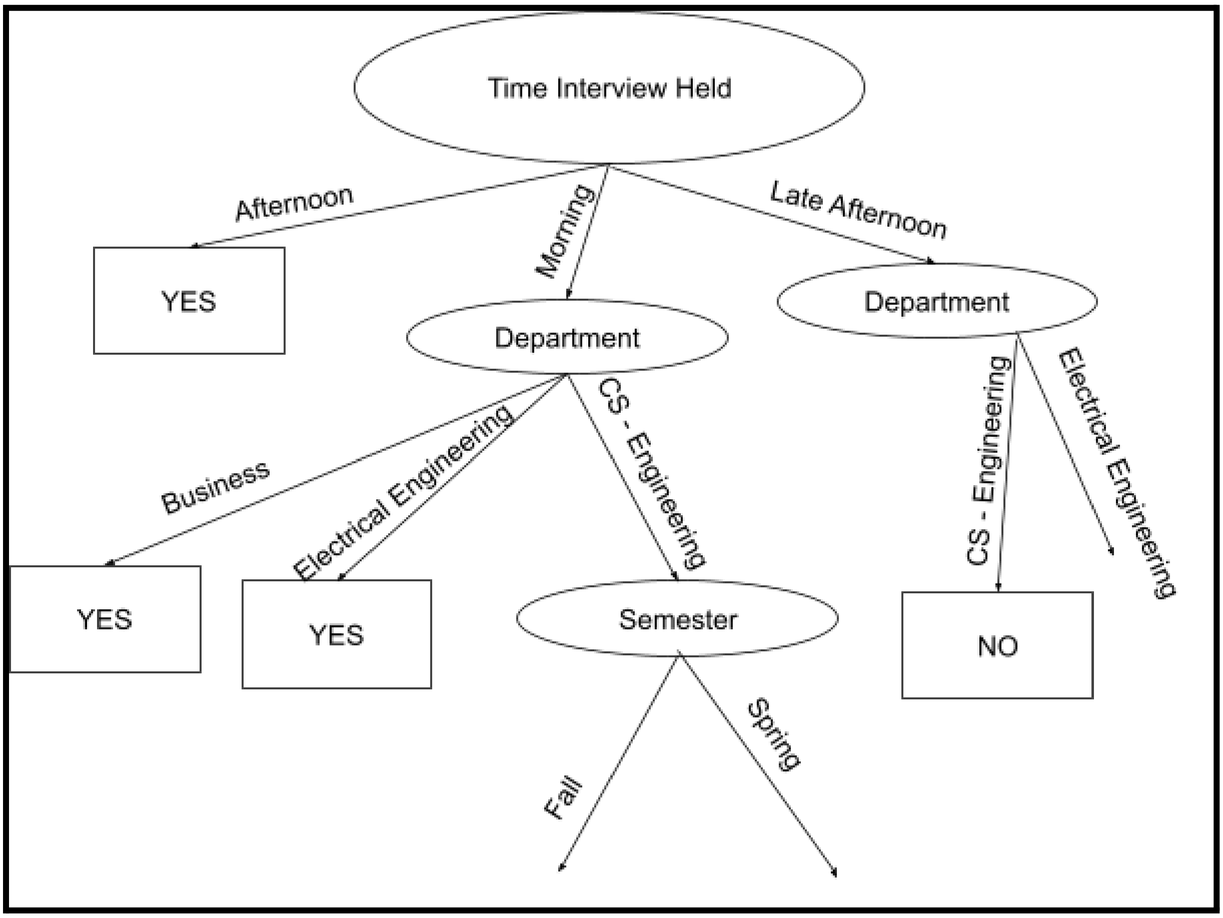
<u>The attribute Semester has the higher information gain and will be selected as the next node.</u>



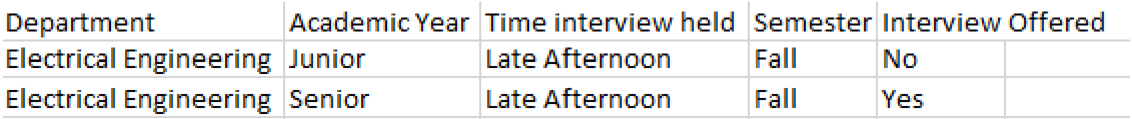
When Time = Late Afternoon and Department = CS - Engineering, output label is No and hence we set leaf node as NO



Department	Academic Year	Time interview held	Semester	Interview Offered
CS - Engineering	Senior	Late Afternoon	Fall	No
CS - Engineering	Sophomore	Late Afternoon	Spring	No



<u>Time = Late Afternoon and Department = Electrical Engineering</u>



Department	Academic Year	Time interview held	Semester	Interview Offered	
Electrical Engineering	Junior	Late Afternoon	Fall	No	
Electrical Engineering	Senior	Late Afternoon	Fall	Yes	

H(Interview Offered | TI = Late Afternoon and Department = Electrical Engineering)
= $-\ (1/2)\ log\ _2(1/2)\ -\ (1/2)\ log_2\ (1/2)$
= **1**

Academic Year

H(Academic Year = Junior) =
-P(Interview Offered = Yes | Academic Year= Junior) $log\ _2$ P(Interview Offered = Yes | Academic Year= Junior) -P(Interview Offered = No | Academic Year= Junior) $log\ _2$ P(Interview Offered = No | Academic Year= Junior)
$-\ 0\ -\ (1)\ log_2\ (1)$
= **0**

H(Academic Year = Senior) =
-P(Interview Offered = Yes | Academic Year= Senior) $log\ _2$ P(Interview Offered = Yes | Academic Year= Senior) -P(Interview Offered = No | Academic Year= Senior) $log\ _2$ P(Interview Offered = No | Academic Year= Senior)

$-\ (1)\ log_{2}(1)\ -\ 0$ = **0**

Information Gain (Academic Year)

IG= H(Interview Offered | TI = Late Afternoon and Department = Electrical Engineering) - P(Academic Year= Junior)H(Academic Year= Junior) - P(Academic Year= Senior)H(Academic Year= Senior)

1- (1/2)*(0) - (1/2)*(0)
= **1**

Semester

H(Time = Late Afternoon and Department = Electrical | Semester = Fall**) = 1**
H(Time = Late Afternoon and Department = Electrical | Semester=**Spring) = 0**
Information Gain: 0

Semester

H(Semester = Fall)

-P(Interview Offered = Yes | Semester = Fall) $log\ _{2}$ P(Interview Offered = Yes | Semester = Fall) - P(Interview Offered = No | Semester = Fall) $log\ _{2}$ P(Interview Offered = No | Semester = Fall)
-$-\ (1/2)\ log\ _{2}(1/2)\ -\ (1/2)\ log\ _{2}(1/2)$
= 1

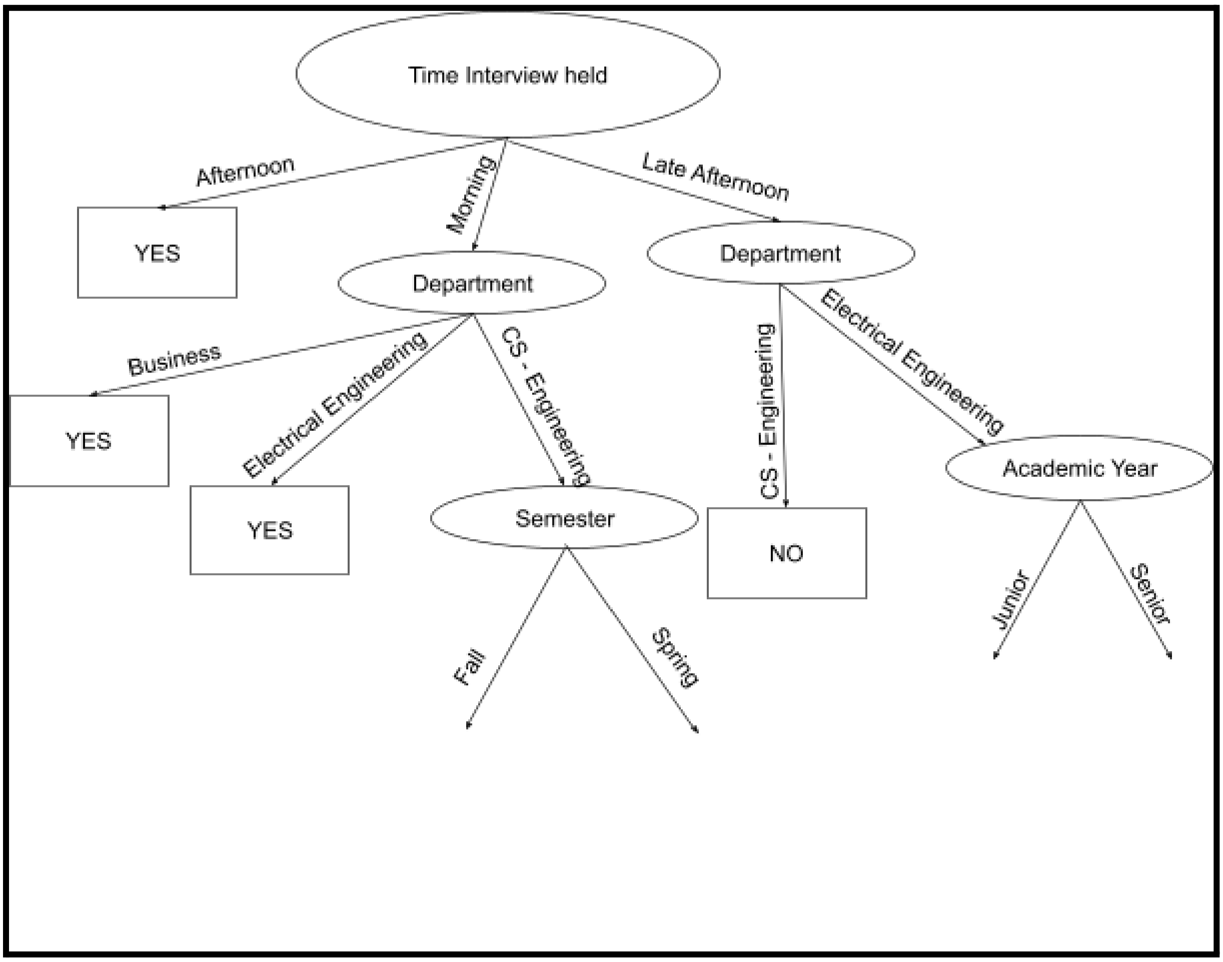
Information Gain (IG Semester)

IG= H(Interview Offered | TI = Late Afternoon and Dept = Electrical Engineering) - P(Semester = Spring)H(Semester = Spring) - P(Semester = Fall)H(Semester = Fall)
= 1-1

= 0

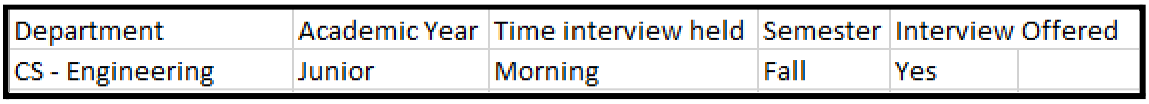
Attribute	**Information Gain**
Academic Year (*)	**1 (*)**
Semester	**0**

<u>Next Selected Root is Academic Year as it has the higher information gain. It has two attributes: Senior and Junior.</u>

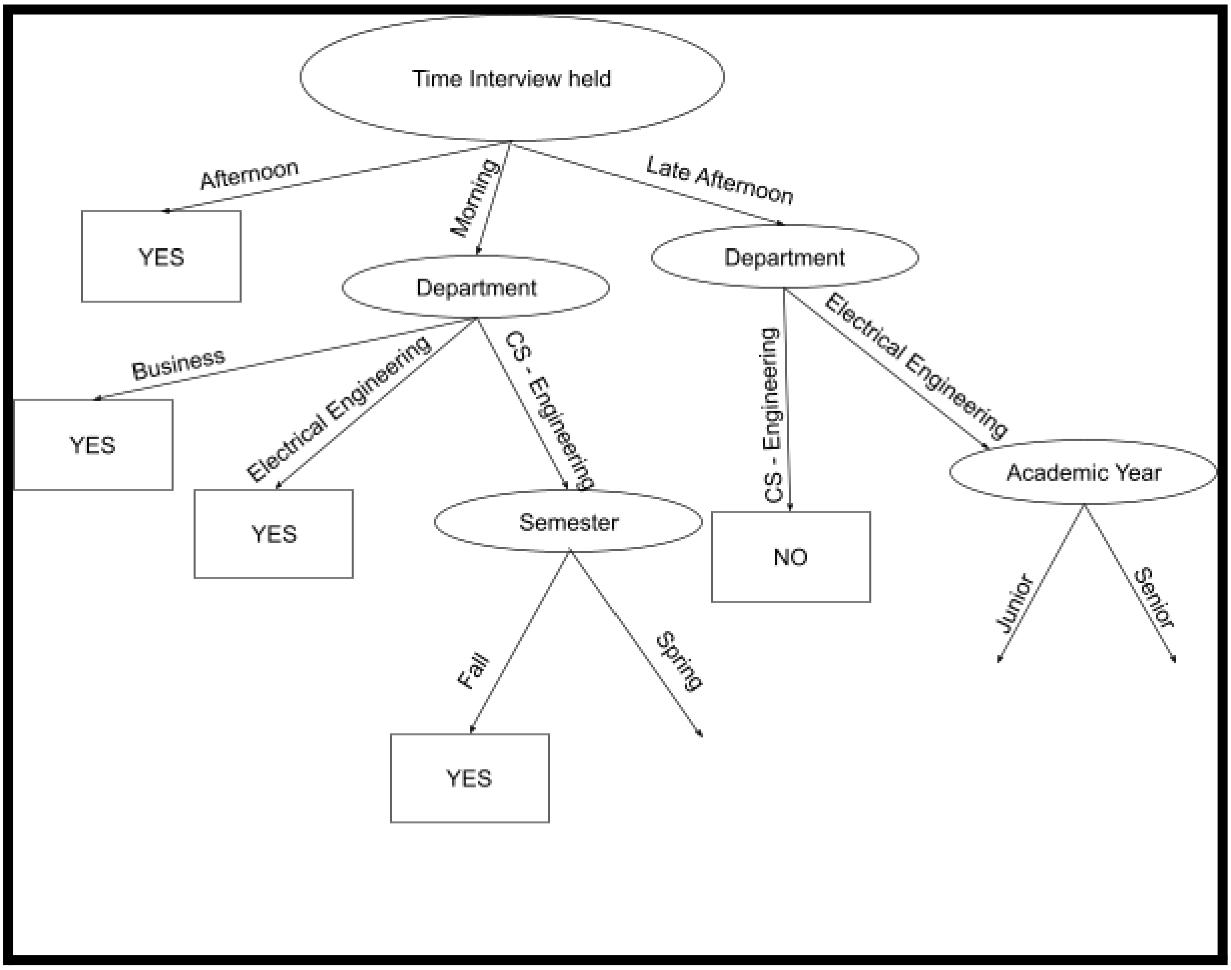


LEVEL 3

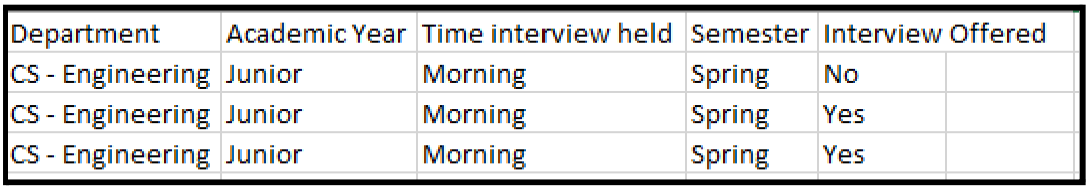
- When Time Interview held = Morning, Department = CS - Engineering and Semester = Fall, we have the class label as Yes. Therefore we have set its leaf node as YES.



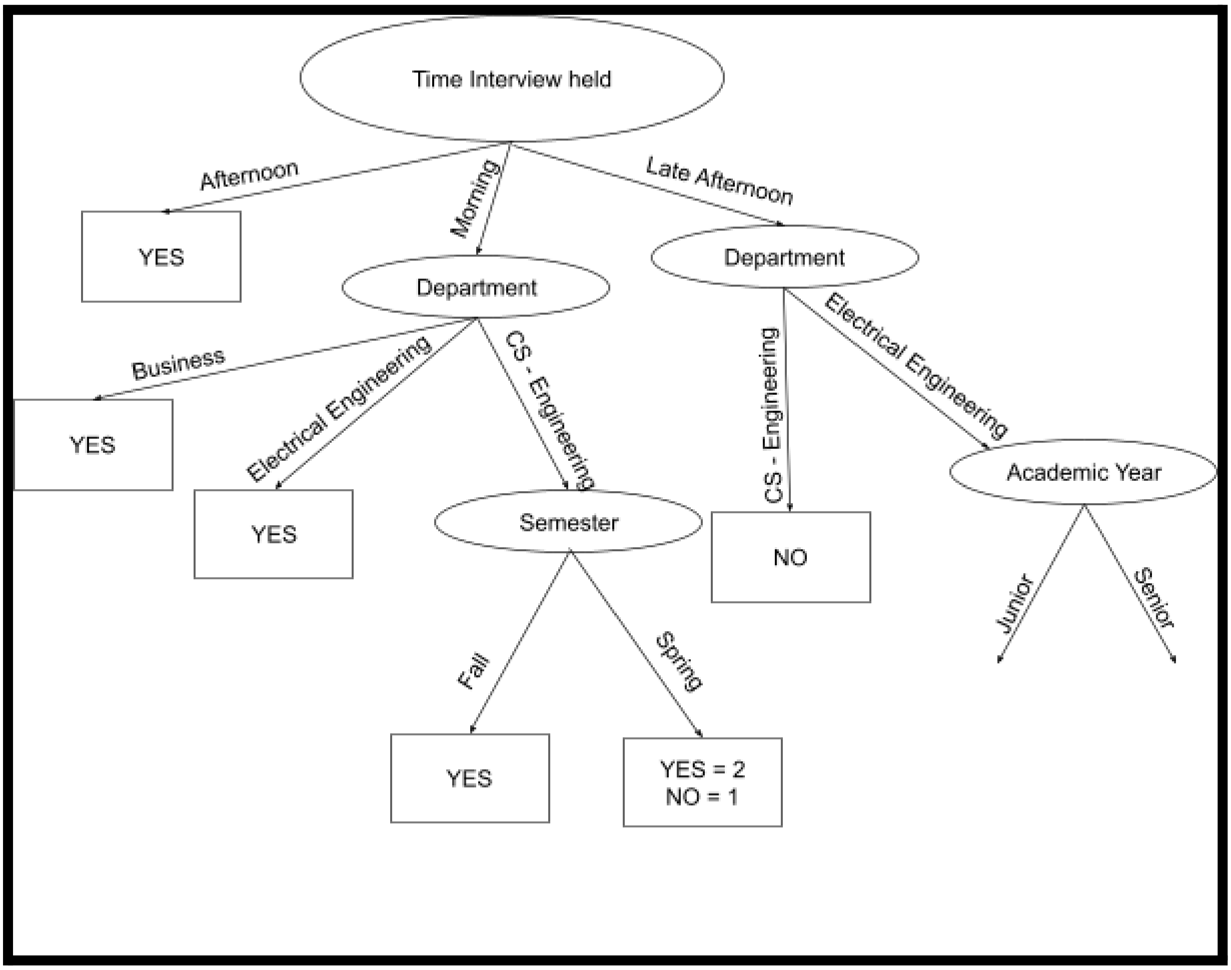
Department	Academic Year	Time interview held	Semester	Interview Offered	
CS - Engineering	Junior	Morning	Fall	Yes	



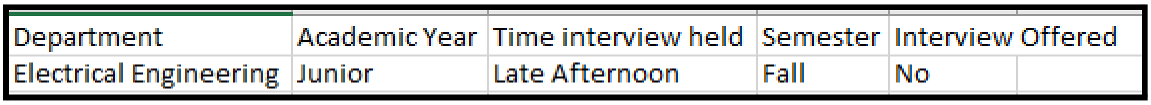
- When Time Interview held = Morning, Department = CS - Engineering and Semester = Spring, we have mixed class labels with 2-Yes and 1-No.



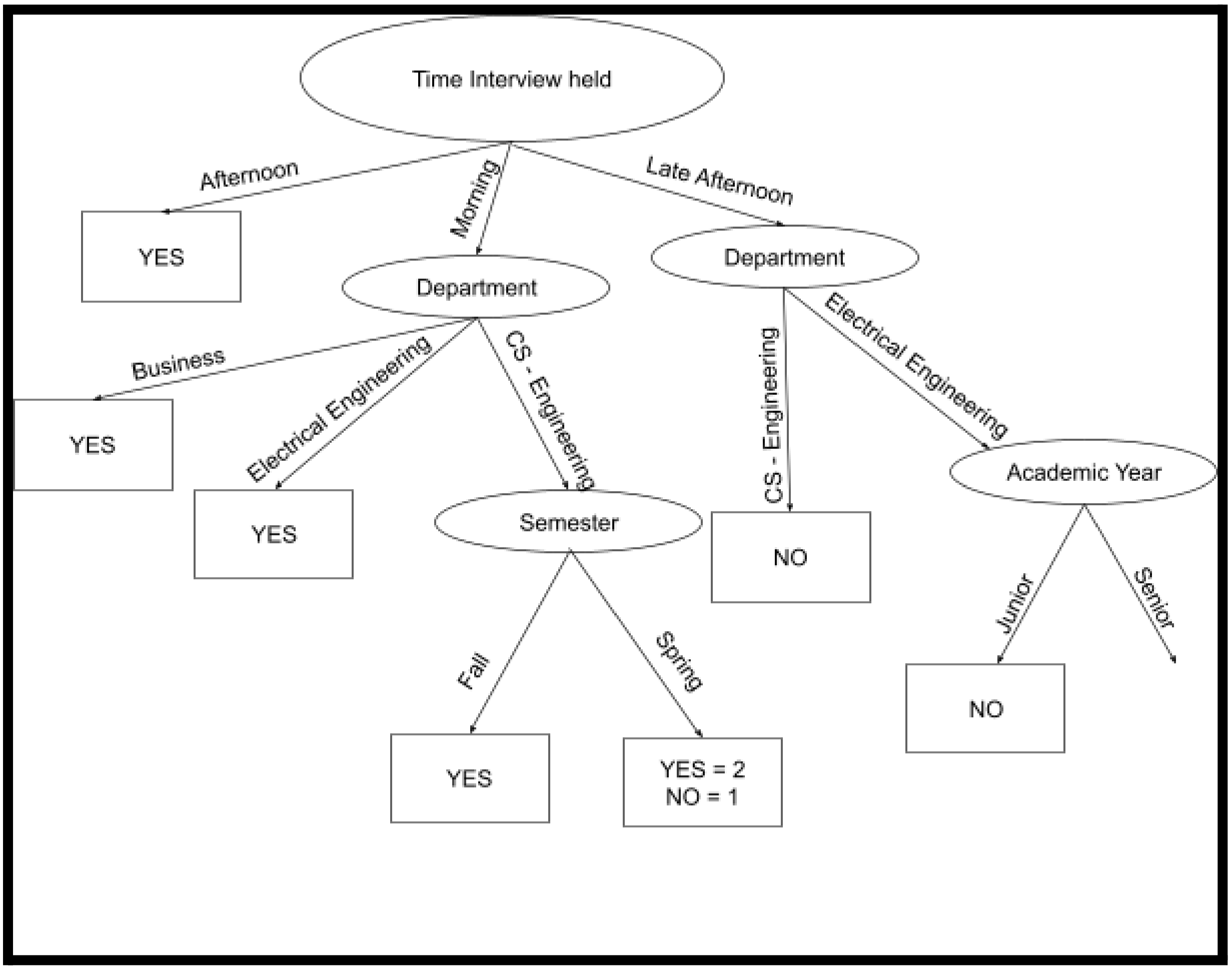
Department	Academic Year	Time interview held	Semester	Interview Offered	
CS - Engineering	Junior	Morning	Spring	No	
CS - Engineering	Junior	Morning	Spring	Yes	
CS - Engineering	Junior	Morning	Spring	Yes	


Time Interview held
Afternoon
Morning
Late Afternoon
YES
Department
Department
Business
Electrical Engineering
CS - Engineering
CS - Engineering
Electrical Engineering
YES
YES
Semester
NO
Academic Year
Junior
Senior
Fall
Spring
YES
YES = 2
NO = 1

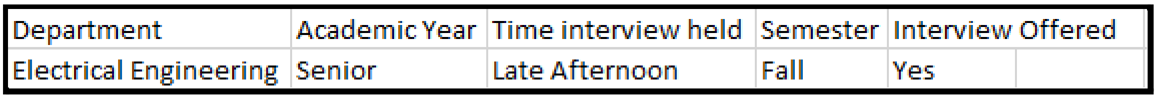
- When Time Interview Held = Late Afternoon, Department = Electrical Engineering and Academic Year = Junior, we have Interview offered = No.



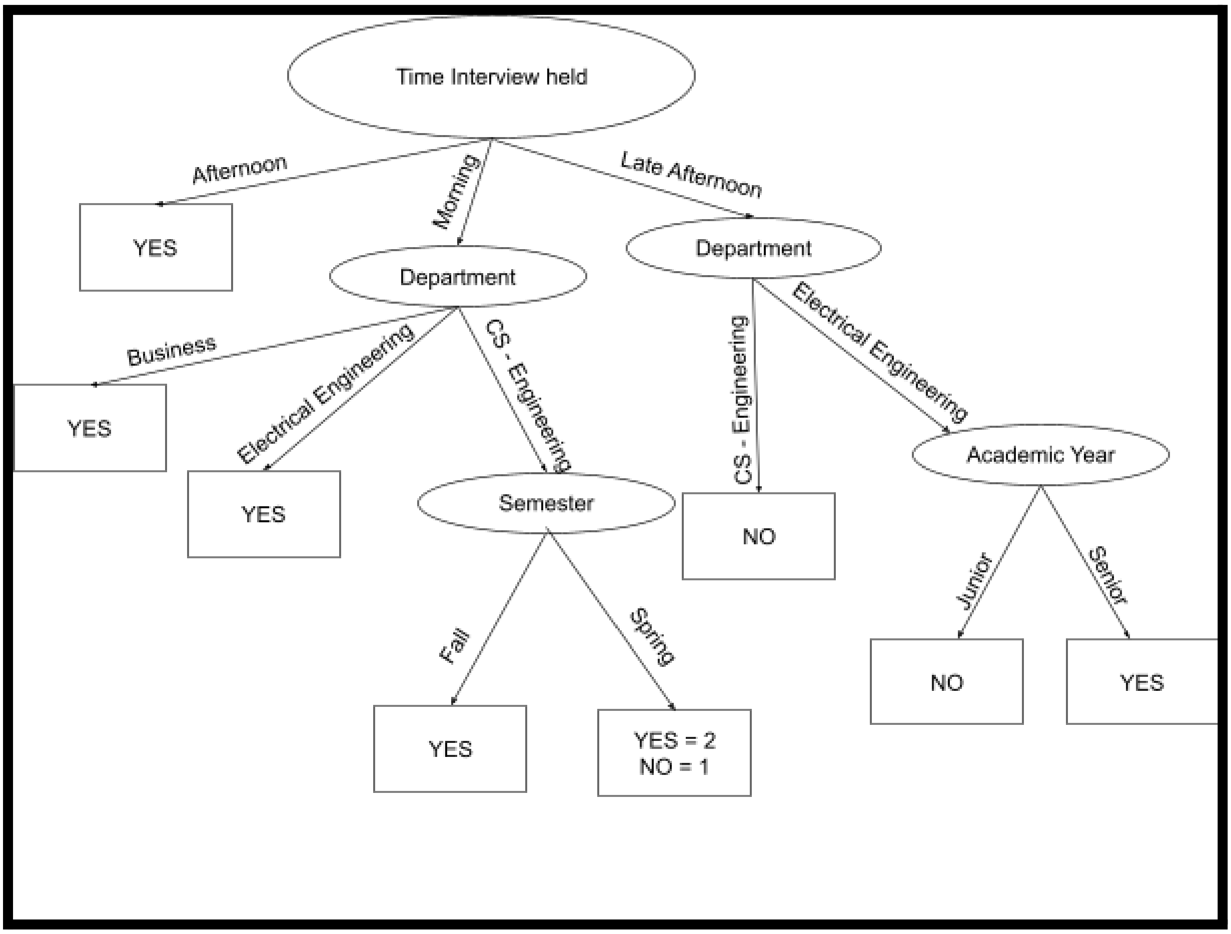
Department	Academic Year	Time interview held	Semester	Interview Offered
Electrical Engineering	Junior	Late Afternoon	Fall	No



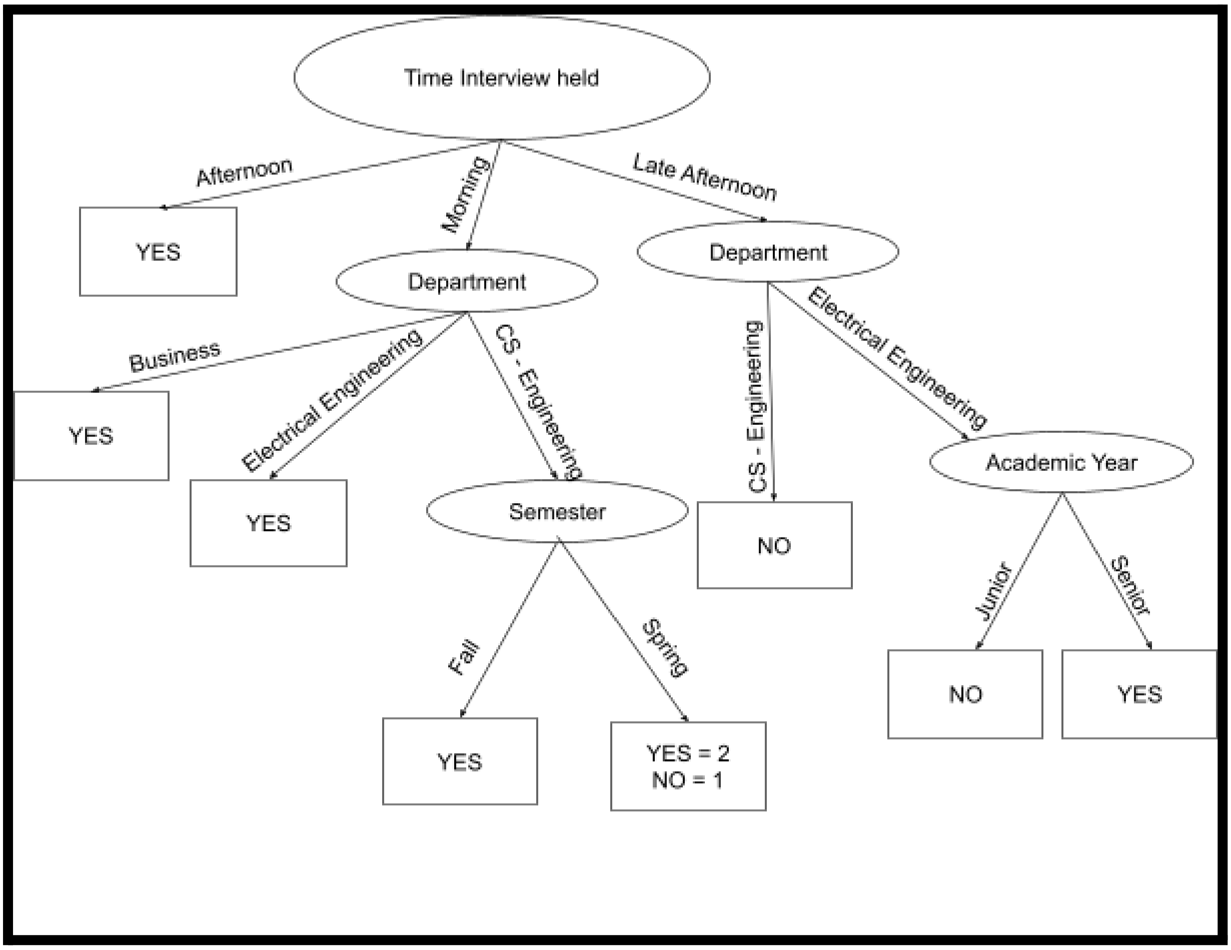
Similarly, when Time Interview Held = Late Afternoon, Department = Electrical Engineering and Academic Year = Senior, we have Interview offered = Yes.



Department	Academic Year	Time interview held	Semester	Interview Offered
Electrical Engineering	Senior	Late Afternoon	Fall	Yes



FINAL DECISION TREE



(b) (15 points) Construct the tree manually using the GINI index, take the attribute academic year as the root (then generate the next two levels of the remaining tree by choosing the best split), write down the computation process by showing the number of cases in each class for each node before splitting and show your tree step by step. (No partial credit)

The question mentions taking the academic year attribute as the root. In the Academic Year attribute, we have three variables, namely, Junior, Sophomore, and Senior.

$$GINI(t) = 1 - \sum_{j} [p(j \mid t)]^2$$

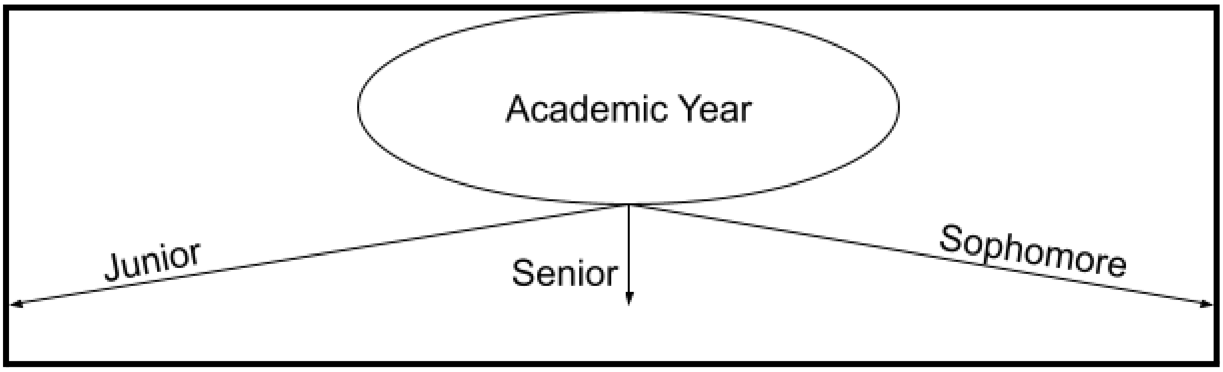
GINI SPLIT

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where n_i *represents the number of records at child i,*

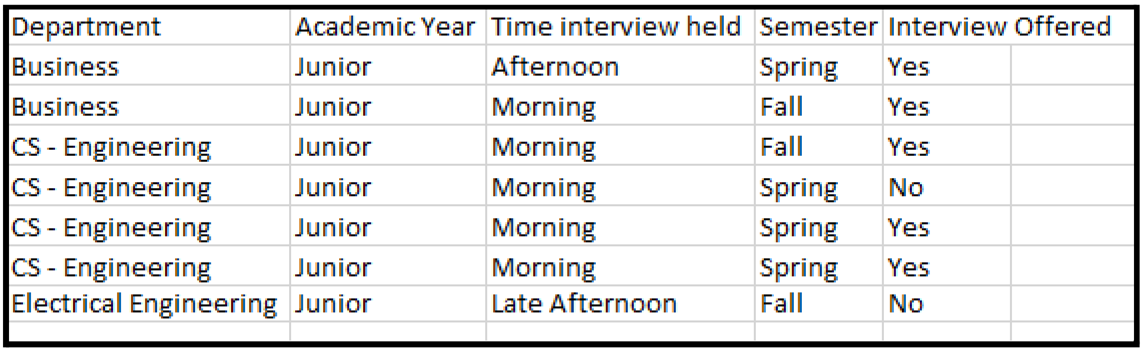
and n *represents the number of records at node* p

Level 0



LEVEL 1

When Academic Year = Junior



Department	Academic Year	Time interview held	Semester	Interview Offered
Business	Junior	Afternoon	Spring	Yes
Business	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Spring	No
CS - Engineering	Junior	Morning	Spring	Yes
CS - Engineering	Junior	Morning	Spring	Yes
Electrical Engineering	Junior	Late Afternoon	Fall	No

Department

$GI(Department\ =\ Business\,|\,Academic\,Year\ =\ Junior)\ =\ 1\ -\ (2/2)^{2}\ -\ (0)^{2}$
1-1= **0**

$GI(Department\ = CS\ -\ Engineering|\,Academic\,Year\ =\ Junior)\ =\ 1\ -\ (3/4)^{2}\ -\ (1/4)^{2}$
=**0.375**

$GI(Department\ =\ Electrical\,|\,Academic\,Year\ =\ Junior)\ =\ 1\ -\ (0)^{2}\ -\ (1)^{2}$
1-1 = **0**

P(Department = Business| Academic Year = Junior) = 2/7
P(Department = CS - Engineering|Academic Year = Junior) = 4/7
P(Department = Electrical Engineering | Academic Year = Junior) = 1/7

GINI SPLIT(Department| AcademicYear=Junior)=2/7(0)+4/7(0.375) + 1/7(0)
0.571*0.375 =**0.214**

Time Interview Held

$GI(Time\ Interview\ Held\ \ =\ \ Afternoon\,|\ \ Academic\,Year\ \ =\ \ Junior)\ =\ 1\ -\ (1/1)^{2}\ -\ (0)^{2}$
1-1 = **0**

$GI(Time\ Interview\ Held\ \ =\ Morning\,|\ \ Academic\,Year\ \ =\ \ Junior)\ =\ 1\ -\ (4/5)^{2}\ -\ (1/5)^{2}$
= **0.32**

$GI(Time\ Interview\ Held\ \ =\ \ Late\ Afternoon\,|\,Academic\,Year\ \ =\ \ Junior)\ =\ 1\ -\ (1/1)^{2}\ -\ (0)^{2}$
1-1 = **0**

P(Time Interview Held = Afternoon| Academic Year = Junior) = 1/7
P(Time Interview Held = Morning|Academic Year = Junior) = 5/7
P(Time Interview Held = Late Afternoon| Academic Year = Junior) = 1/7

GINI SPLIT(Time Interview Held)= 1/7(0) + 5/7(0.32) + 1/7(0)
=0.228

<u>Semester</u>

$GI(Semester\ =\ Fall\,|\,Academic\,Year\ =\ Junior)\ =\ 1\ -\ (2/3)^{2}\ -\ (1/3)^{2}$
1-1 = **0.4444**

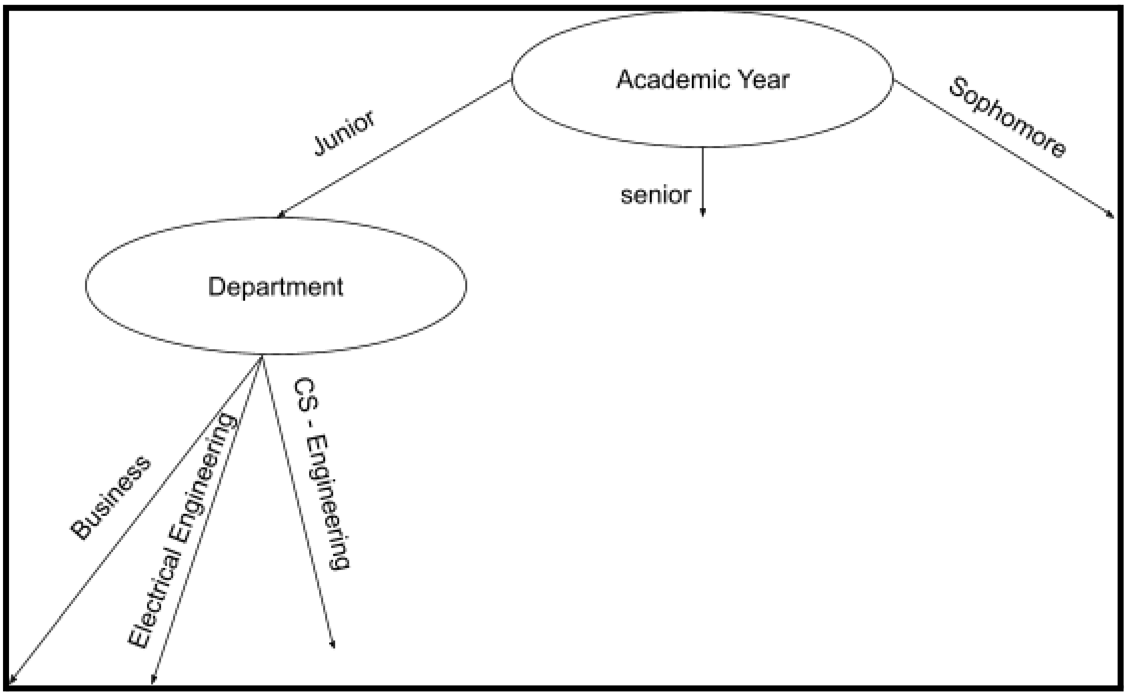
$GI(Semester\ =\ Spring\,|\,Academic\,Year\ =\ Junior)\ =\ 1\ -\ (3/4)^{2}\ -\ (1/4)^{2}$
= **0.375**

P(Semester = Fall| Academic Year = Junior) = 3/7
P(Semester = Spring| Academic Year = Junior) = 4/7

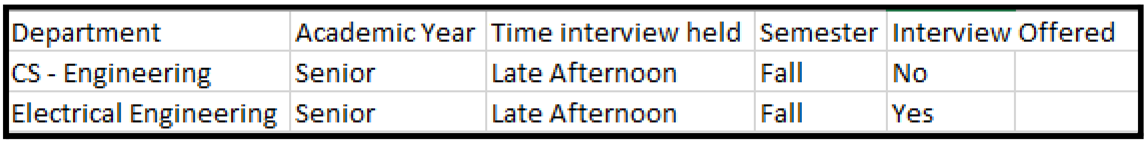
GINI SPLIT(Semester)= 3/7(0.4444) + 4/7(0.375)
=0.404

Attribute	**GINI SPLIT**
<u>Department (*)</u>	***<u>0.214 (*)</u>***
Time Interview Held	**0.228**
Semester	**0.404**

Lowest GINI SPLIT is for Department and it becomes our next node.



<u>When Academic Year = Senior</u>



Department	Academic Year	Time interview held	Semester	Interview Offered
CS - Engineering	Senior	Late Afternoon	Fall	No
Electrical Engineering	Senior	Late Afternoon	Fall	Yes

Department

$GI(Department\ =\ CS\ -\ Engineering\ |\ Academic\ Year\ =\ Senior)\ =\ 1\ -\ 0\ -\ (1/1)^2$
=**0**

$GI(Department\ =\ Electrical\ Engineering\ |\ Academic\ Year\ =\ Senior)\ =\ 1\ -\ (1)^2\ -\ (0)^2$
1-1 = **0**

P(Department = CS - Engineering| Academic Year = Senior) = 1/2
P(Department = Electrical Engineering| Academic Year = Senior) = 1/2

GINI SPLIT(Department) =
(1/2) * (0) + (1/2)*(0)
= 0

Time Interview Held

$GI(Time\ Interview\ Held\ =\ Late\ Afternoon\ |\ Academic\ Year\ =\ Senior)\ =\ 1\ -\ (1/2)^2\ -\ (1/2)^2$
1-0.5 = **0.5**

P(Time Interview Held = Late Afternoon|Academic Year = Senior) = 2/2=1
GINI SPLIT(Time Interview Held)= 1*(0.5)
=0.5

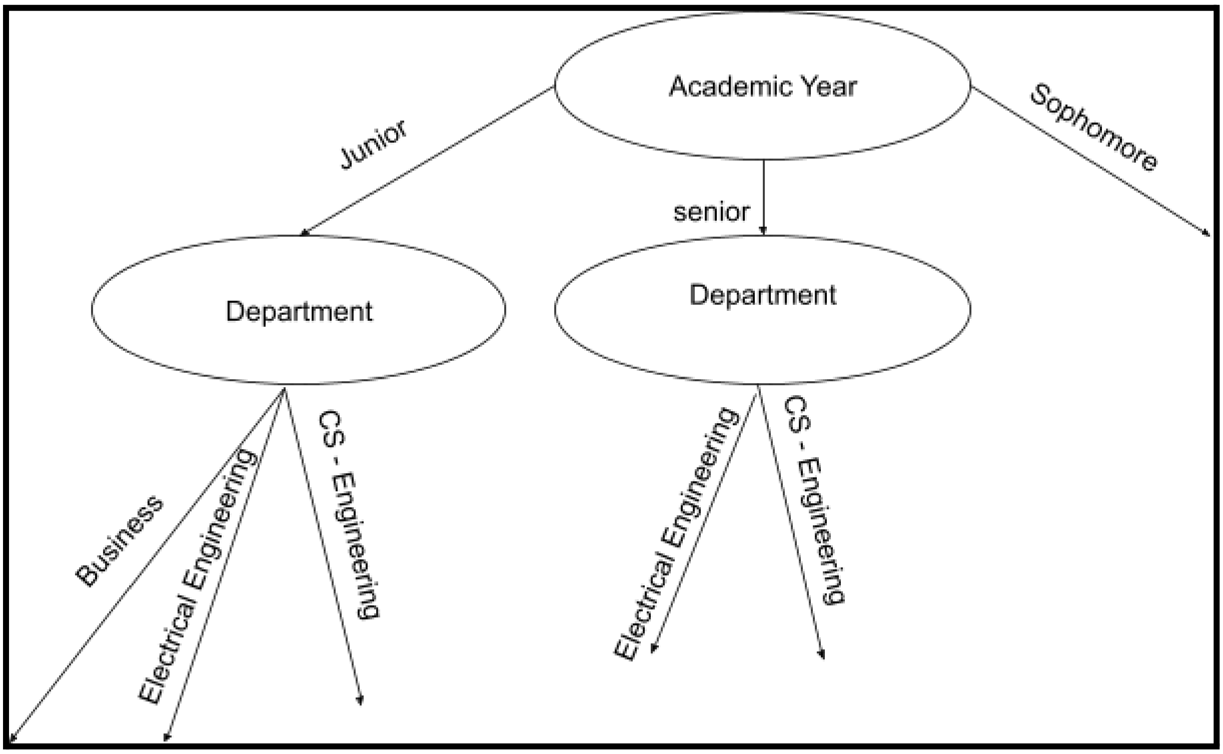
Semester

$GI(Semester\ =\ Fall\,|\ |\,Academic\,Year\ =\ Senior)\ =\ 1\ -\ (1/2)^{2}\ -\ (1/2)^{2}$
1-0.5
= **0.5**

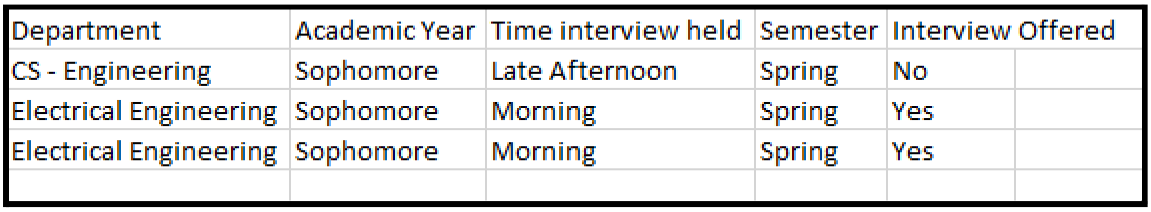
P(Semester = Fall| Academic Year = Senior) = 2/2=1

GINI SPLIT(Semester)= P(Semester = Fall) * GI(Semester = Fall)
= (2/2)*0.5
= 0.5

Attribute	**GINI SPLIT**
Department (*)	***0***
Time Interview Held	**0.5**
Semester	**0.5**



<u>When Academic Year = Sophomore</u>



Department	Academic Year	Time interview held	Semester	Interview Offered	
CS - Engineering	Sophomore	Late Afternoon	Spring	No	
Electrical Engineering	Sophomore	Morning	Spring	Yes	
Electrical Engineering	Sophomore	Morning	Spring	Yes	

Department

$GI(Department\ =\ CS\ -\ Engineering\,|\,Academic\ Year\ =\ Sophomore)\ =\ 1\ -\ 0\ -\ (1/1)^{2}$
=0

$GI(Department\ =\ Electrical\ Engineering\,|\,Academic\ Year\ =\ Sophomore)\ =\ 1\ -\ (2/2)^{2}\ -\ (0)^{2}$
1-1 = **0**

P(Department = CS - Engineering | Academic Year = Sophomore) = 1/3
P(Department = Electrical Engineering|Academic Year = Sophomore) = 2/3

GINI SPLIT (Department) = P(Department = CS - Engineering) * GI(Department = CS - Engineering) + P(Department = Electrical Engineering) * GI(Department = Electrical Engineering)
(1/3)*(0) + (2/3)*(0)
= 0

Time Interview Held

$GI(\ Time\ Interview\ Held\ =\ Late\ Afternoon\ |\ Academic\ Year\ =\ Sophomore)\ =\ 1\ -\ (0)^{2}\ -\ (1/1)^{2}$
1-1 = **0**

$GI(\ Time\ Interview\ Held\ =\ Morning\ |\ Academic\ Year\ =\ Sophomore)\ =\ 1\ -\ (2/2)^{2}\ -\ (0)^{2}$
1-1 = **0**

P($Time\ Interview\ Held\ =\ Late\ Afternoon\ |\ Academic\ Year\ =\ Sophomore$**) = 1/3**

P($Time\ Interview\ Held\ =\ Morning\ |\ Academic\ Year\ =\ Sophomore$**) = 2/3**

GINI SPLIT(Time Interview Held)=
P(Time Interview Held = Late Afternoon) * GI(Time Interview Held = Late Afternoon) +
P(Time Interview Held =Morning) * GI(Time Interview Held = Morning)

(1/3)*(0) + (2/3)*(0)
= 0

Semester

$GI(Semester\ =\ Spring\ |\ Academic\ Year\ =\ Sophomore)\ =\ 1\ -\ (2/3)^{2}\ -\ (1/3)^{2}$
1-(4/9) - (1/9) = **0.444**

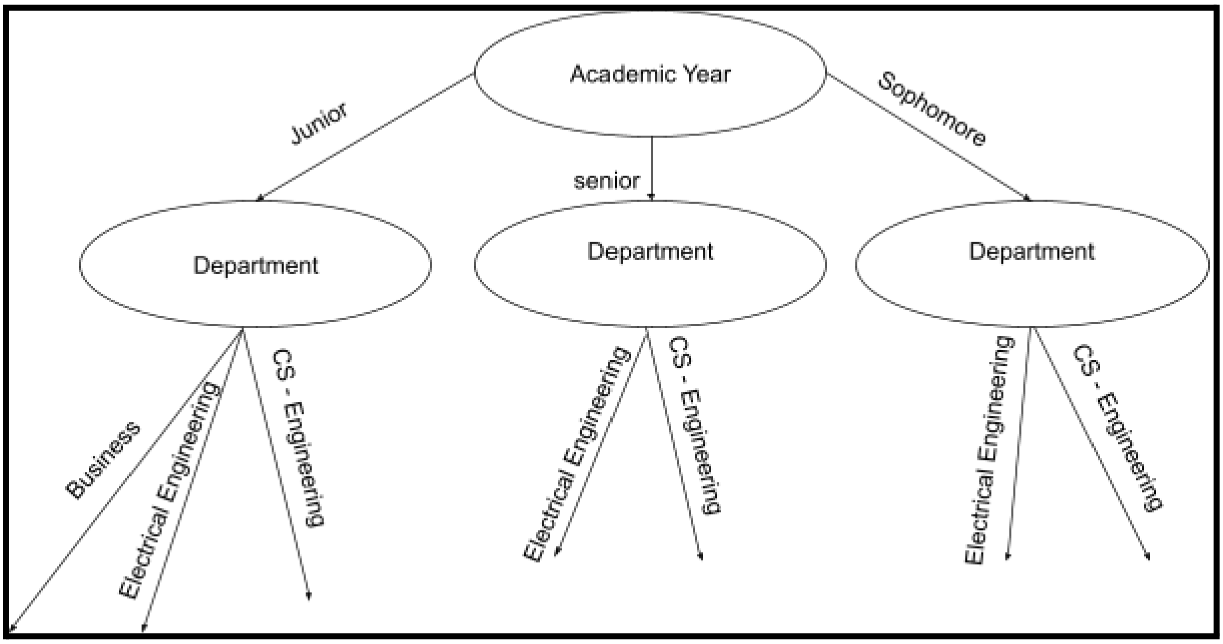
P($Semester\ =\ Spring\ |\ Academic\ Year\ =\ Sophomore$**) = 3/3=1**

GINI SPLIT(Semester)=
P(Semester = Spring) * GI(Semester = Spring)

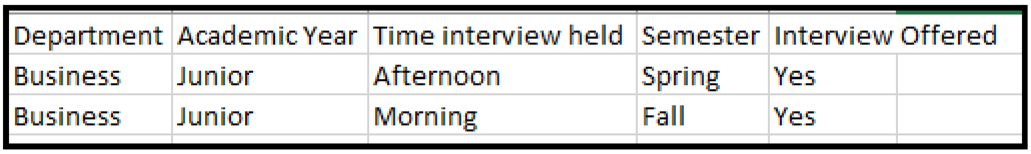
(0.444)*(1)
=0.444

Attribute	**GINI SPLIT**
Department (*)	***0 (*)***
Time Interview Held	**0**
Semester	**0.444**

The problem statement states that incase of a tie, we are required to break the tie in favour of the left-most attribute. In our set, we have the Department attribute in the left-most attribute.

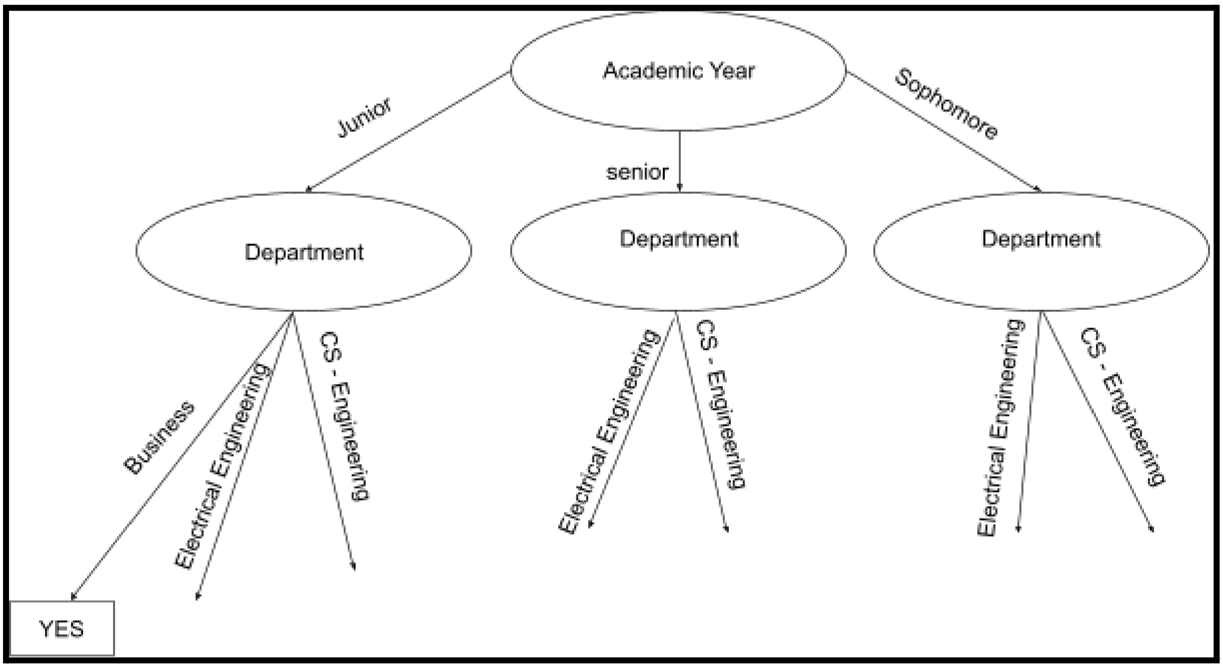


When Academic Year = Junior and Department = Business

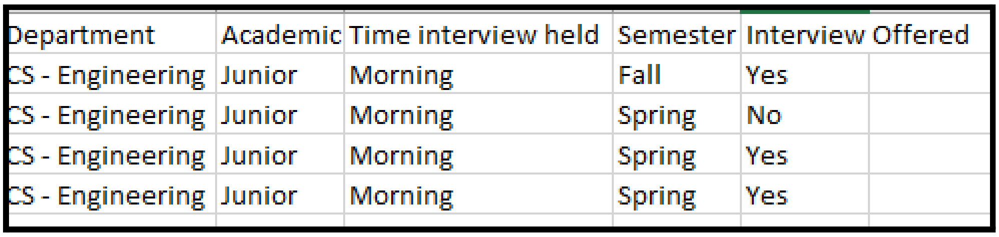


Department	Academic Year	Time interview held	Semester	Interview Offered
Business	Junior	Afternoon	Spring	Yes
Business	Junior	Morning	Fall	Yes

We see that our output class label is Yes and hence we set our leaf node as YES



<u>When Academic Year = Junior and Department = CS-Engineering</u>



Department	Academic	Time interview held	Semester	Interview Offered
CS - Engineering	Junior	Morning	Fall	Yes
CS - Engineering	Junior	Morning	Spring	No
CS - Engineering	Junior	Morning	Spring	Yes
CS - Engineering	Junior	Morning	Spring	Yes

<u>Time Interview Held</u>

$GI(Academic\ Year\ =\ Junior\ and\ Department\ =\ CS\ -\ Engineering\ and\ Time\ interview\ held\ =\ Morning)$

$1\ -\ (3/4)^{2}\ -\ (1/4)^{2}$

= 6/16

= **0.375**

GINI SPLIT(Time Interview Held)=

P(Time Interview Held = Morning) * GI (Time Interview Held = Morning)

(1)*0.375 **= 0.375**

<u>Semester</u>

$GI(Academic\ Year\ =\ Junior\ and\ Department\ =\ CS\ -\ Engineering\ and\ Semester\ =\ Fall)\ =\ 1\ -\ (1)^{2}\ -$

= **0**

$GI(Academic\ Year\ =\ Junior\ and\ Department\ =\ CS\ -\ Engineering\ and\ Semester\ =\ Spring)\ \ =\ \ 1\ -\ (2/3$

= **0.444**

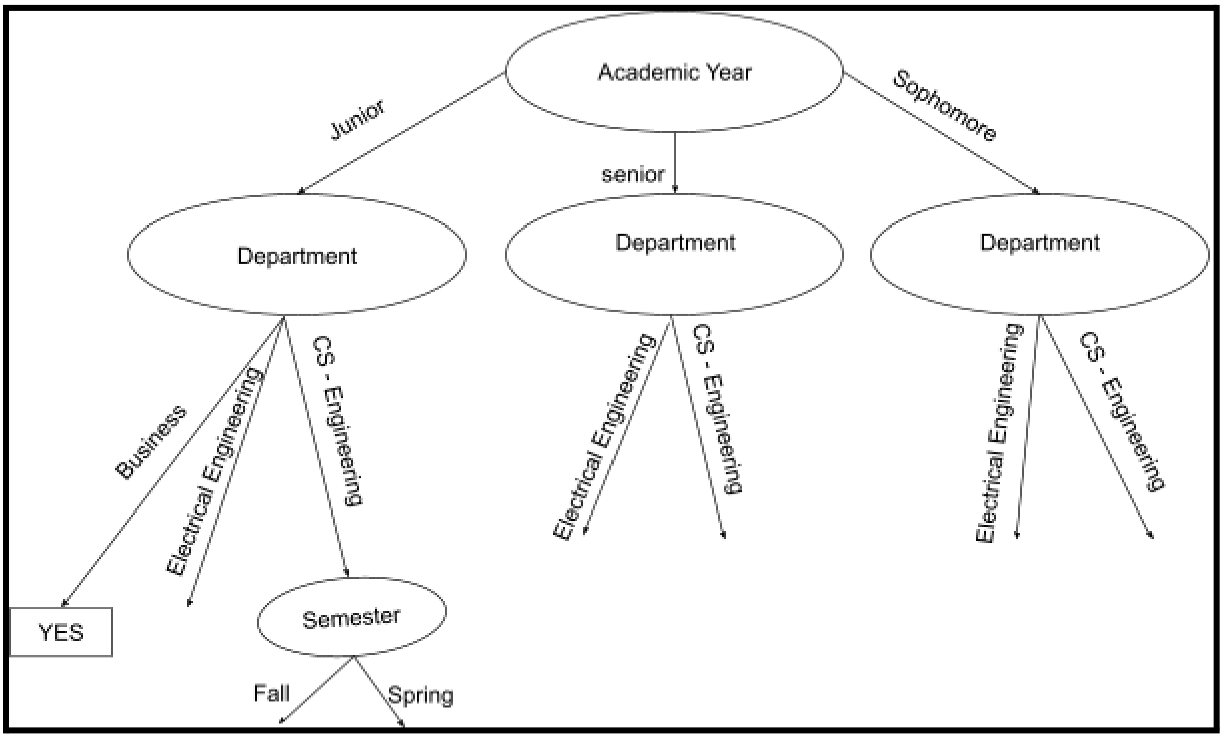
GINI SPLITSemester) =

P(Semester = Spring) * GI(Semester = Spring)

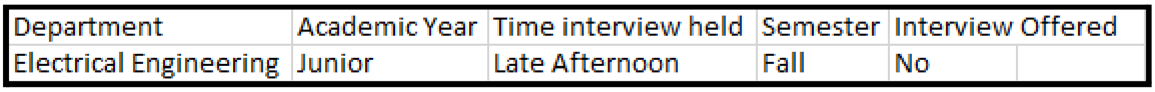
(1/4)*(0) + (3/4)*(0.444) = 0.333

Attribute	**GINI SPLIT**
Time Interview Held	**0.375**
Semester (*)	**0.333 (*)**

Semester has the lowest GINI Index and is selected as our next node. It has two attributes, namely, Fall and Spring.

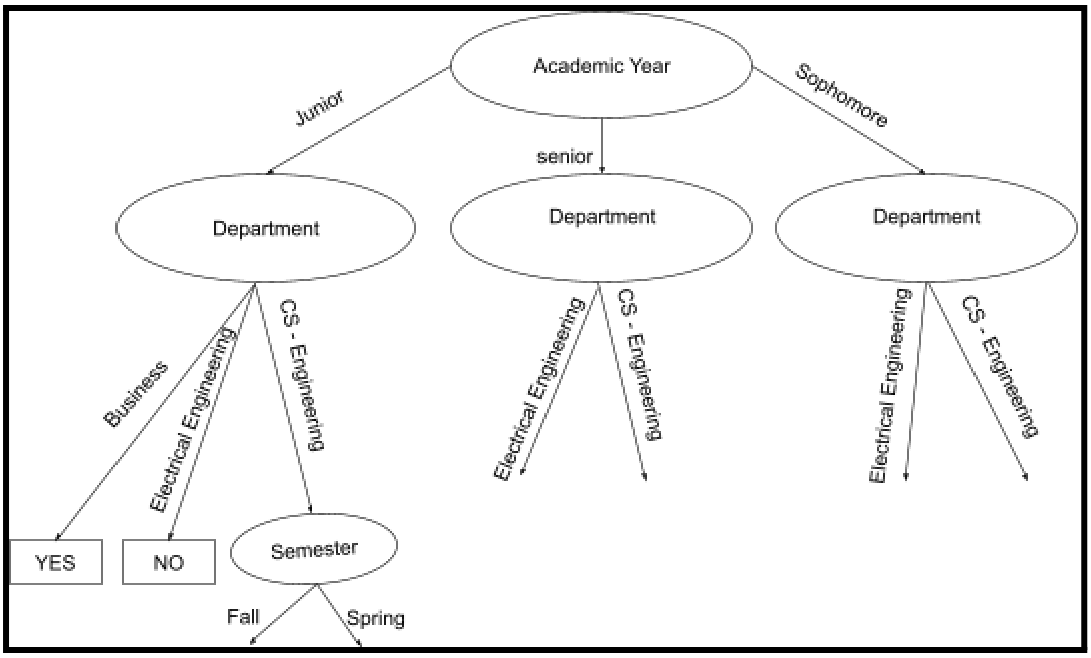


When Academic Year = Junior and Department = Electrical Engineering

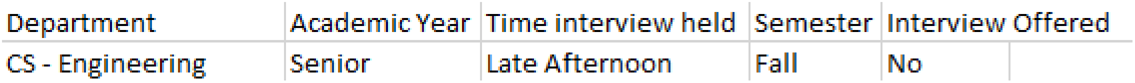


Department	Academic Year	Time interview held	Semester	Interview Offered
Electrical Engineering	Junior	Late Afternoon	Fall	No

The output class label is No. We therefore set its leaf node as No.

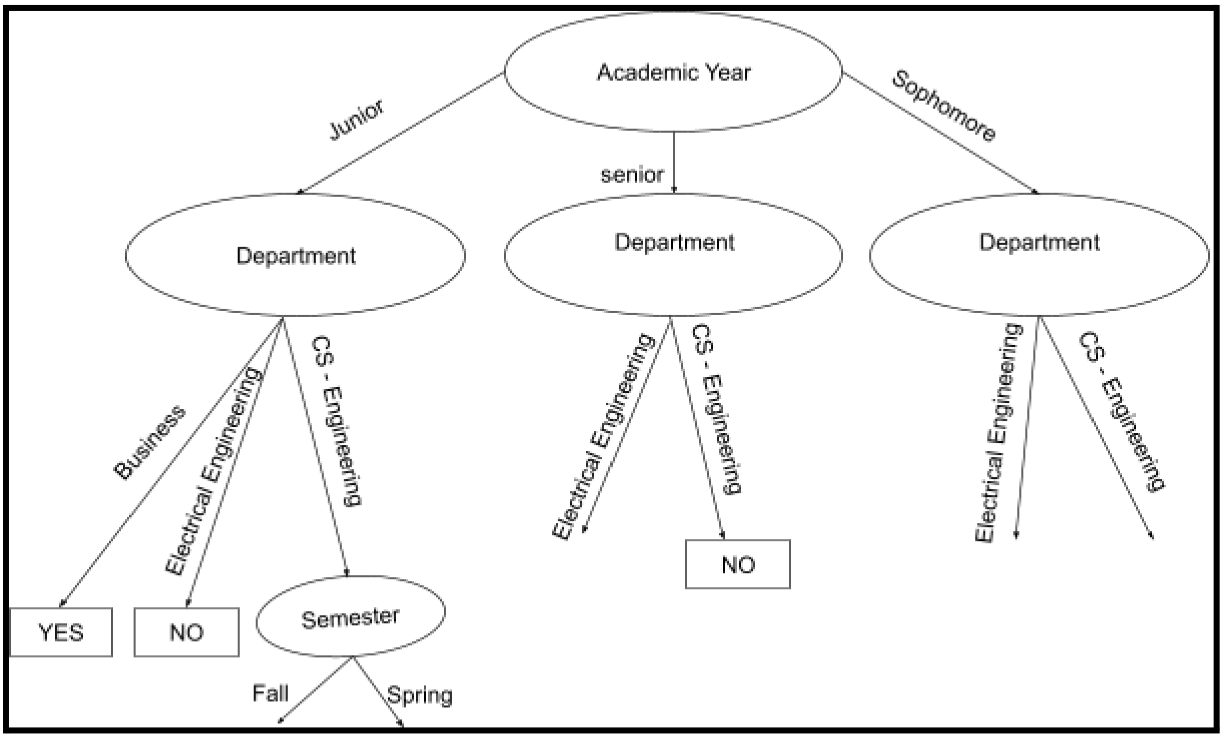


When Academic Year = Senior and Department = CS - Engineering

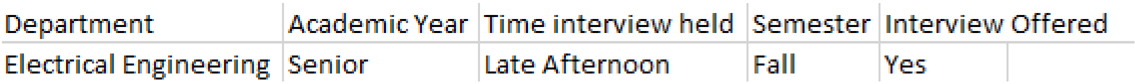


Department	Academic Year	Time interview held	Semester	Interview Offered
CS - Engineering	Senior	Late Afternoon	Fall	No

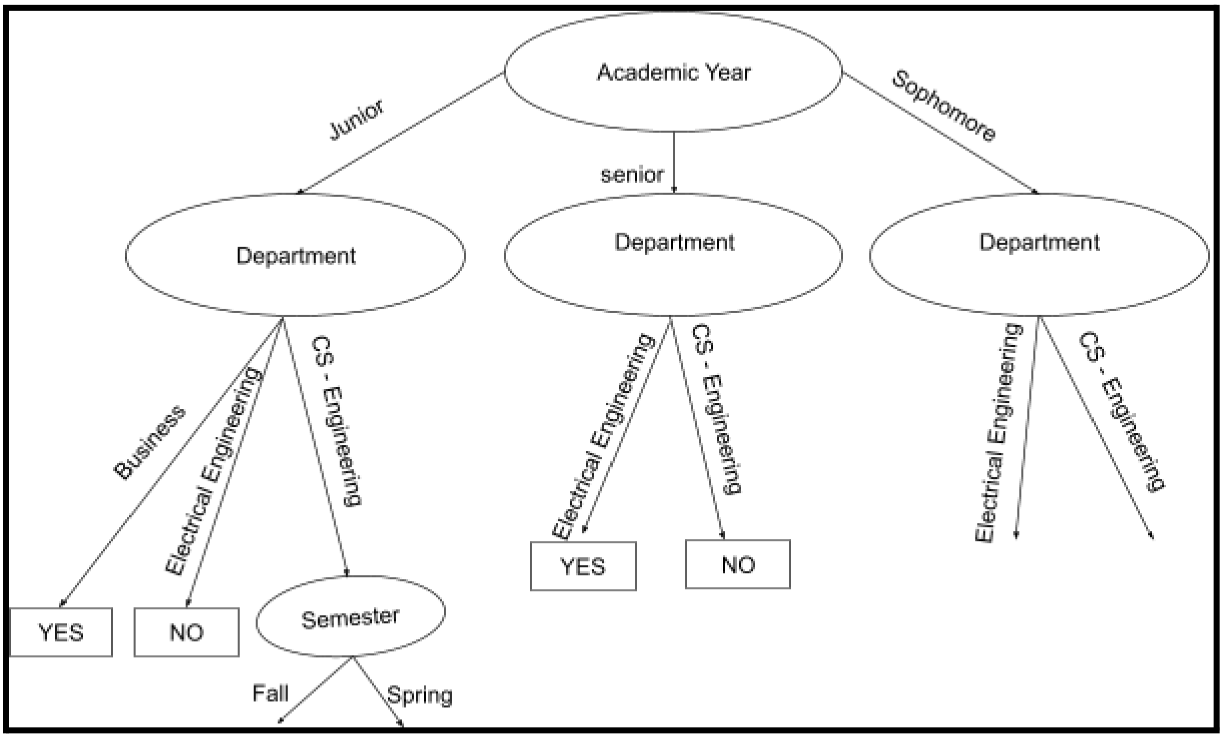
In this case, we have our output variable as No. Hence we set our leaf node as NO.



<u>When Academic Year = Senior and Department = Electrical Engineering</u>

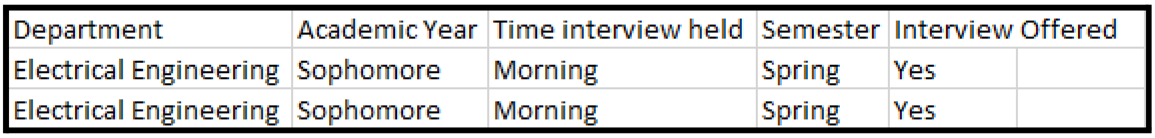


Department	Academic Year	Time interview held	Semester	Interview Offered
Electrical Engineering	Senior	Late Afternoon	Fall	Yes



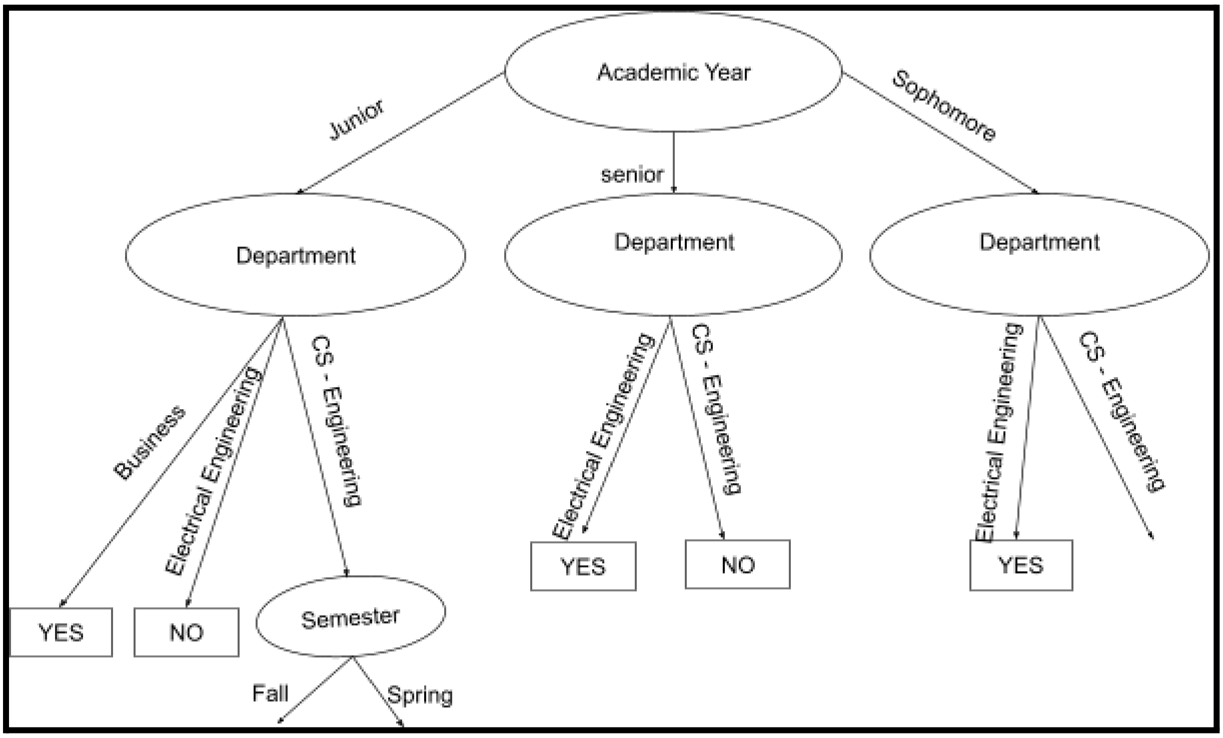
In this case, we have our output variable as YES. Hence we set our leaf node When Academic Year = Senior and Department = Electrical Engineering as YES.

When Academic Year = Sophomore and Department = Electrical Engineering

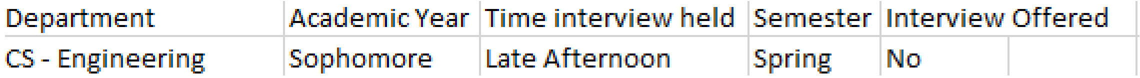


Department	Academic Year	Time interview held	Semester	Interview Offered
Electrical Engineering	Sophomore	Morning	Spring	Yes
Electrical Engineering	Sophomore	Morning	Spring	Yes

In this case, we have our output variables as YES. Hence we set our leaf node when Academic Year = Sophomore and Department = Electrical Engineering as YES.

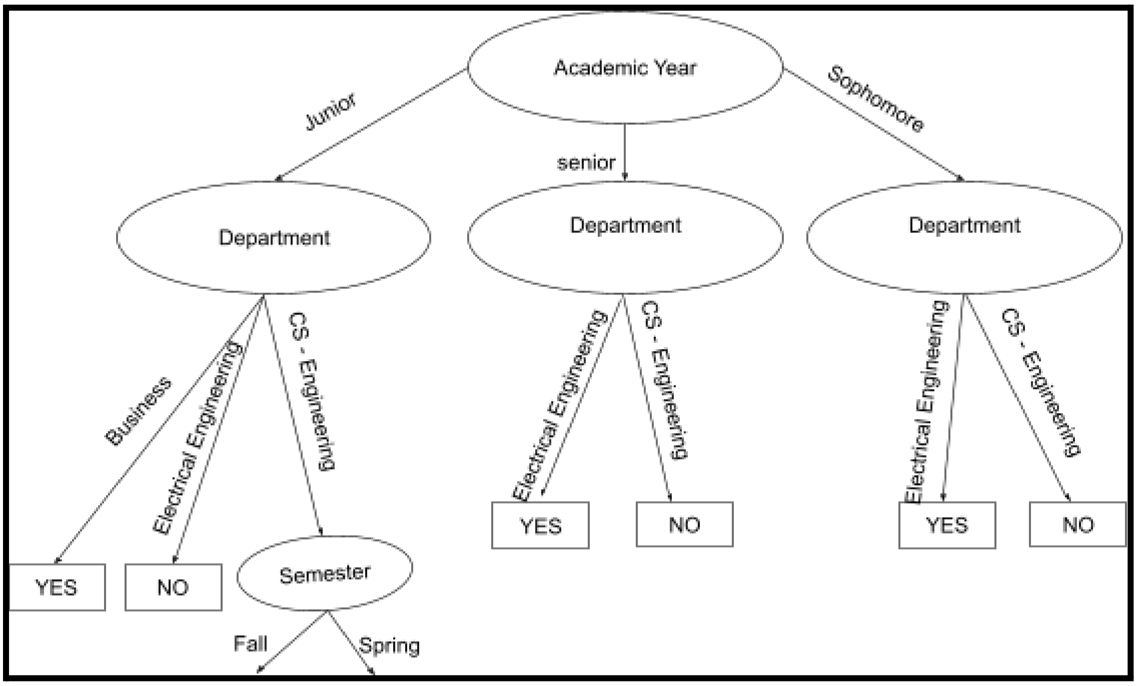


When Academic Year = Sophomore and Department = CS- Engineering

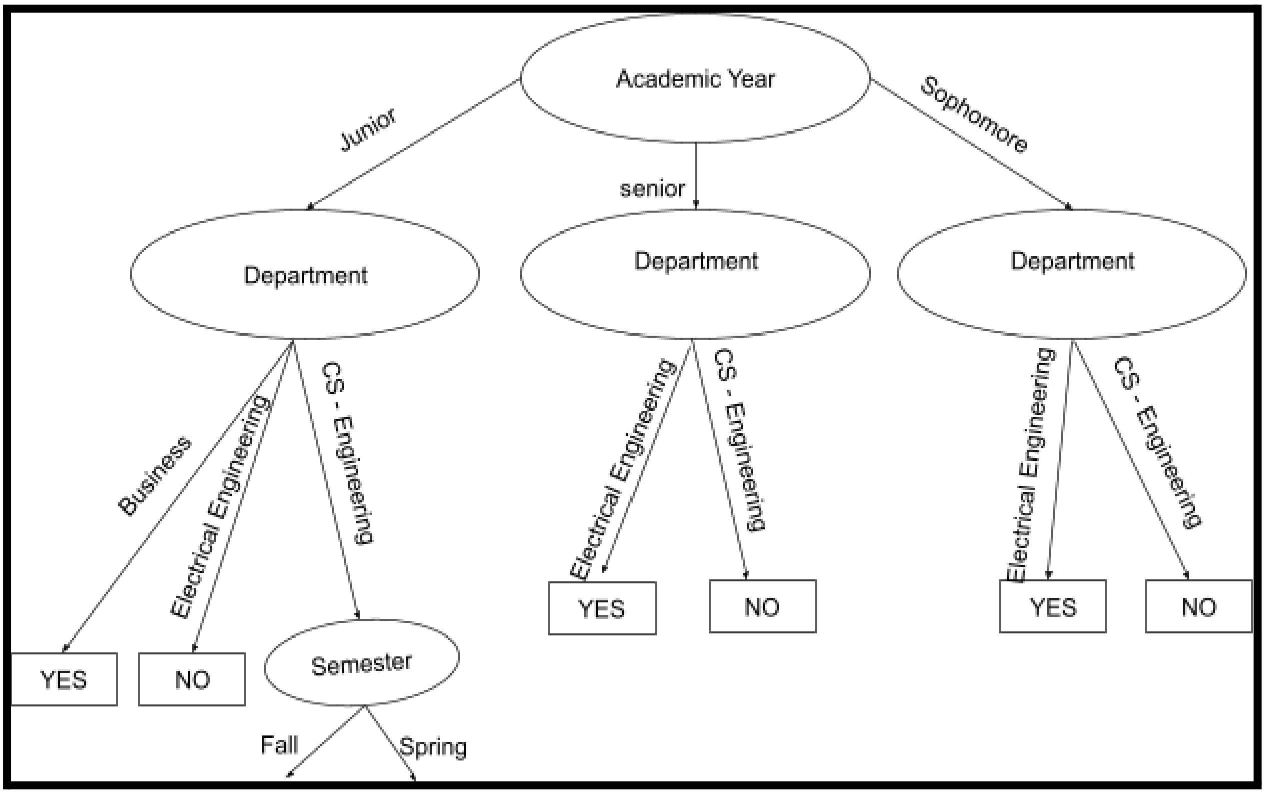


Department	Academic Year	Time interview held	Semester	Interview Offered
CS - Engineering	Sophomore	Late Afternoon	Spring	No

In this case, we have our output variable as NO. Hence we set our leaf node when Academic Year = Sophomore and Department = CS- Engineering as NO.



Final Decision Tree is :



3. (30 points) [Evaluate Classier] [Angela Zhang] Joe pays careful attention to the weather. Unfortunately, the weather report is not reliable. It frequently predicts rain or storm incorrectly. Joe decides to create his own decision tree to help him stay dry. Figure 1 below is the decision tree created by Joe (yes if it will rain; no if it will not rain). Your task is to determine whether to prune the given decision tree. Additionally, you will use the provided test dataset in \RainPredict.csv" to determine the effectiveness of resulted decision trees.

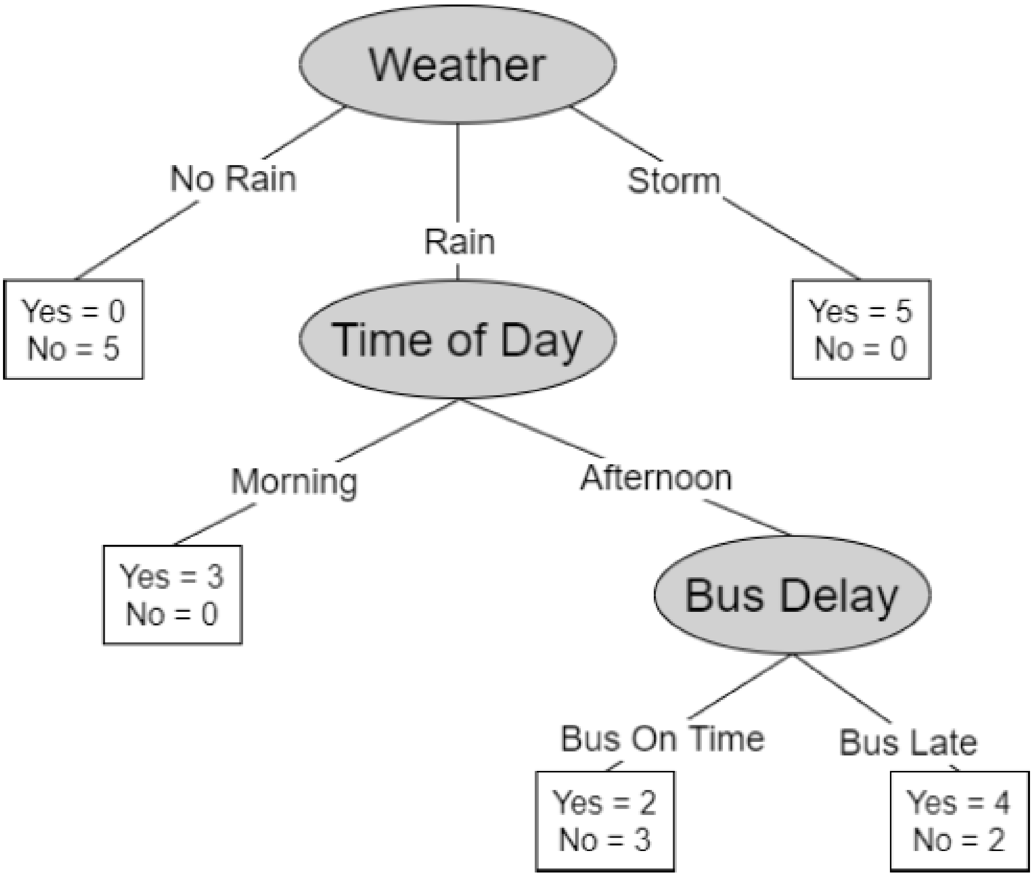


Figure 1: The Rain Prediction decision tree.

(a) (10 points) Post-pruning based on optimistic errors.

i. (3 points) Calculate the optimistic errors before splitting and after splitting using the attribute Bus Delay respectively.

- **Before splitting** the ‘Bus Delay’ node in the decision tree, the class label values of the instance are,

Yes = 4 + 2 = 6 &

No = 3 + 2 = 5

So Training error (optimistic) for this node is: **5/11**

- **After splitting** the 'Bus Delay' node into two leaf nodes of values,

  Yes = 2 and No = 3

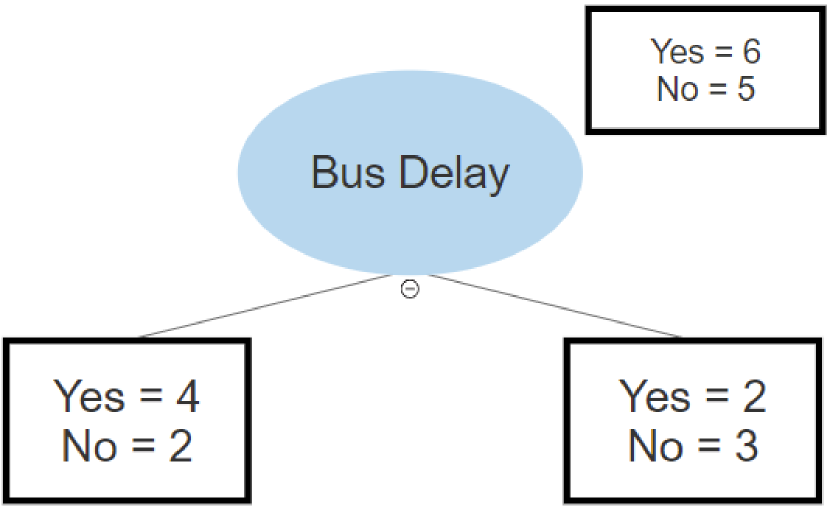
  &

  Yes = 4 and No = 2

  So Training error (optimistic) for these nodes is:

  (2 + 2)/11 = **4/11**

- So the optimistic training error for attribute 'Bus Delay' for **before** and **after** splitting of the node is **5/11** and **4/11** respectively.

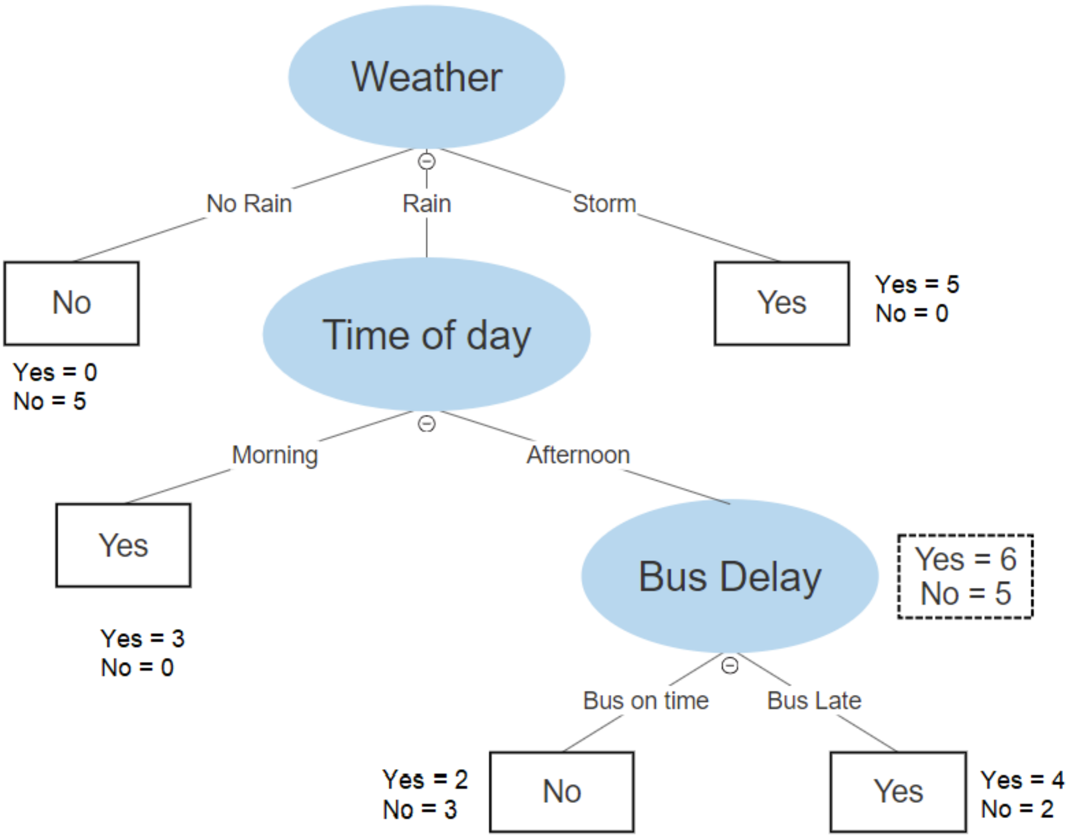


ii. (2 points) Based upon the optimistic errors, would the subtree be pruned or retained? If it is pruned, draw the resulting decision tree and use it for the next question; otherwise, use the original decision tree shown in Figure 1 for the next question.

- We know that the optimistic training error for attribute 'Bus Delay' for before and after splitting of the node is **5/11** and **4/11** respectively.

- Since 5/11 > 4/11 we need not prune the tree at the node of this attribute, since the leaves have a lower error than the parent node itself.

So the tree would be retained and remains the same,



iii. (5 points) Use the decision tree from (a).ii above to classify the test dataset (RainPredict.csv). Report its performance on the following five evaluation metrics: Accuracy, Recall (Sensitivity), Precision, Specificity, and F1 Measure.

Based on the decision tree above the predicted output for the data set “RainPredict.csv” is:

Weather	**Time of Day**	**Bus Delay**	**Get caught in rain?** **(Actual Class - y)**	**Predicted Class - ŷ**
No Rain	Morning	Not Late	No	No
No Rain	Afternoon	Not Late	No	No

No Rain	Afternoon	Late	No	No
Rain	Morning	Not Late	Yes	Yes
Rain	Morning	Late	Yes	Yes
Rain	Afternoon	Late	Yes	Yes
Rain	Afternoon	Late	Yes	Yes
Rain	Afternoon	Not Late	Yes	No
Rain	Morning	Late	No	Yes
Rain	Afternoon	Not Late	No	No
Rain	Afternoon	Not Late	No	No
Storm	Afternoon	Late	No	Yes
Storm	Afternoon	Not Late	Yes	Yes
Storm	Morning	Not Late	No	Yes

From above the confusion matrix can be calculated:

	PREDICTED CLASS (ŷ)		
ACTUAL CLASS (y)		**Class = YES**	**Class = NO**
	Class = YES	TP (a) = 5	FN (b) = 1
	Class = NO	FP (c) = 3	TN (d) = 5

- **Accuracy**:

  The formula for accuracy is: $(TP + TN) / (TP + FN + FP + TN)$
  = (5 + 5) / (5 + 1 + 3 + 5) = 10 / 14 = **0.714**

- **Recall**:

  The formula for recall is: $(TP) / (TP + FN)$
  = (5) / (5 + 1) = 5 / 6 = **0.833**

- **Precision**:

  The formula for precision is: $(TP) / (TP + FP)$
  = (5) / (5 + 3) = 5 / 8 = **0.625**

- **Specificity**:

  The formula for precision is: $(TN)\ /\ (TN\ +\ FP)$
  = (5) / (5 + 3) = 5 / 8 = **0.625**

- **F1-measure**:

  The formula for F-measure is: $(2rp)\ /\ (r\ +\ p)$
  = (2 * 0.833 * 0.625) / (0.833 + 0.625) = (1.04125) / (1.458)
  = **0.714**

(b) (10 points) Post-pruning based on pessimistic errors. When calculating pessimistic errors, each leaf node will add a factor of 2 to the error.

i. (3 points) Calculate the pessimistic errors before splitting and after splitting using the attribute Bus Delay respectively.

- For **pessimistic error** calculation, an additional factor of **2** is taken for each leaf node.

- **Before splitting** the 'Bus Delay' node in the decision tree, the class label values of the instance are,

  Yes = 4 + 2 = 6 &

  No = 3 + 2 = 5

  So pessimistic error for this node is: (5 + 2*1)/ 11 = **7/11**

- **After splitting** the 'Bus Delay' node into two leaf nodes of values,

  Yes = 2 and No = 3

  &

  Yes = 4 and No = 2

  So Training error (optimistic) for these nodes is:

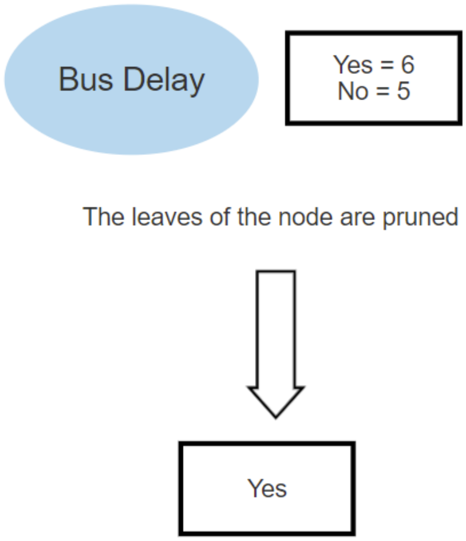
  (2 + 2)/11 = (4 + 2*2)/11 = **8/11**

- So the pessimistic error for attribute ‘Bus Delay’ for **before** and **after** splitting of the node is **7/11** and **8/11** respectively.

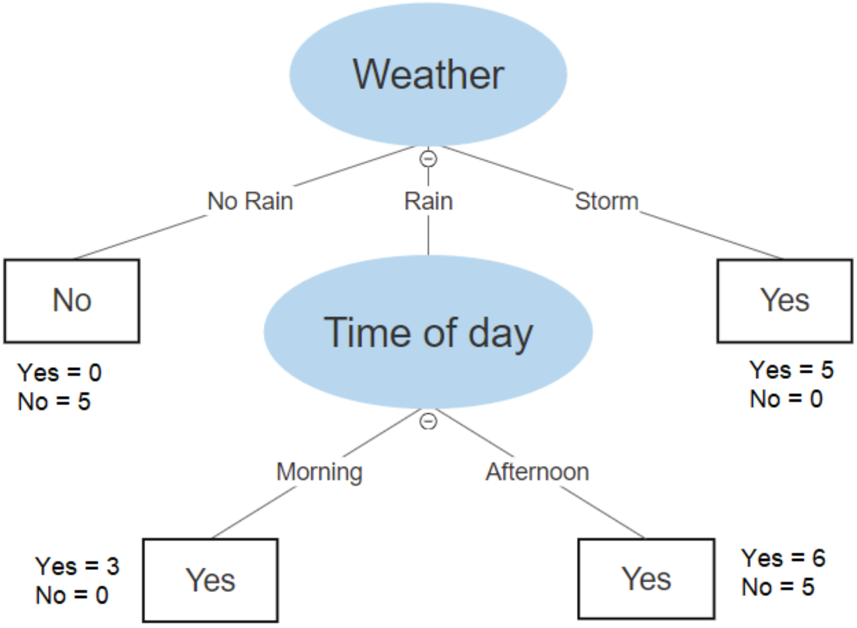
ii. (2 points) Based on the pessimistic errors, would the subtree be pruned or retained? If it is pruned, draw the resulting decision tree and use it for the next question; otherwise, use the original decision tree shown in Figure 1 for the next question.

- We know that the pessimistic error for attribute ‘Bus Delay’ for **before** and **after** splitting of the node is **7/11** and **8/11** respectively.

- Since 7/11 > 8/11, we need to prune the subtree at the node representing this attribute, since the leaves have a higher error than the parent node itself. By pruning the leaves we have a much stable tree due to low error.



The new tree after pruning is represented as:



iii. (5 points) Use the decision tree from (b).ii above to classify the test dataset (RainPredict.csv). Report its corresponding five evaluation metrics: Accuracy, Recall(Sensitivity), Precision, Specificity, and F1 Measure.

Based on the newly pruned decision tree above the predicted output for the data set “RainPredict.csv” this time will be:

Weather	**Time of Day**	**Bus Delay**	**Get caught in rain? (Actual Class - y)**	**Predicted Class - ŷ**
No Rain	Morning	Not Late	No	No
No Rain	Afternoon	Not Late	No	No
No Rain	Afternoon	Late	No	No
Rain	Morning	Not Late	Yes	Yes
Rain	Morning	Late	Yes	Yes
Rain	Afternoon	Late	Yes	Yes
Rain	Afternoon	Late	Yes	Yes

Rain	Afternoon	Not Late	Yes	Yes
Rain	Morning	Late	No	Yes
Rain	Afternoon	Not Late	No	Yes
Rain	Afternoon	Not Late	No	Yes
Storm	Afternoon	Late	No	Yes
Storm	Afternoon	Not Late	Yes	Yes
Storm	Morning	Not Late	No	Yes

	PREDICTED CLASS (ŷ)		
ACTUAL CLASS (y)		**Class = YES**	**Class = NO**
	Class = YES	TP (a) = 6	FN (b) = 0
	Class = NO	FP (c) = 5	TN (d) = 3

- **<u>Accuracy</u>**:

  The formula for accuracy is: $(TP + TN) / (TP + FN + FP + TN)$
  = (6 + 3) / (6 + 0 + 5 + 3) = 9 / 14 = **0.642**

- **<u>Recall</u>**:

  The formula for recall is: $(TP) / (TP + FN)$
  = (6) / (6 + 0) = 6 / 6 = **1**

- **<u>Precision</u>**:

  The formula for precision is: $(TP) / (TP + FP)$
  = (6) / (6 + 5) = 6 / 11 = **0.545**

- **Specificity**:

  The formula for precision is: $(TN) / (TN + FP)$
  = (3) / (3 + 5) = 3 / 8 = **0.375**

- **F1-measure**:

  The formula for F-measure is: $(2rp) / (r + p)$
  = (2 * 1 * 0.545) / (1 + 0.545) = (1.09) / (1.545)
  = **0.705**

(c) (10 points) We will compare the performance of the decision trees from (a).ii and from (b).ii using the test dataset (RainPredict.csv). For the task of predicting if Joe will get caught in the rain, which of the five evaluation metrics: Accuracy, Recall(Sensitivity),
Precision, Specificity, and F1 Measure, are the most important? Based on your selected evaluation metrics, which decision tree, (a).ii or (b).ii, is better for this task? Justify your answers.

Comparing the Accuracy, Recall, Precision, Specificity, and F1-measure for both subtree in (a).ii and (b).ii:

Evaluation metrics	**For the decision tree in (a).ii**	**For the decision tree in (b).ii**
Accuracy	0.714	0.642
Recall	**0.833**	**1**
Precision	0.625	0.545
Specificity	0.625	0.375
F1-measure	0.714	0.705

- We can see that the values of the evaluation metrics for the second tree have dropped compared to the values of the first tree, except for **Recall**.

- The accuracy of the decision tree in (a).ii is higher than the second tree, but accuracy is not the best metric to compare both trees. Accuracy is a good measure in models where having a False Positive (FP) is much more expensive than having a False Negative (FN) value, which might be useful for detecting credit card fraud or evaluating a website blocker. But not in this case.

- Here we are trying to predict rainfall based on various attributes. In this model having a False positive (FP) value is not very expensive, i.e, Joe might incorrectly predict that it might rain from his decision tree but it might not rain, and it is okay if it doesn't. But on the other hand, False Negatives (FN) are costly, i.e, Joe incorrectly predicts that it won't rain, but it will.

- The **Recall** is the metric that evaluates the ratio of correctly predicted positive observations to all observations in actual positive class. It takes into consideration the False Negative (FN) values and hence is the perfect evaluation metric we can use to pick the better decision tree.

- Referring to the confusion matrix from the (b).ii tree above, the number of FN is 0. Moreover, the recall value of the second tree is higher than the first. These values make a good case for our conclusion that the **decision tree in (b).ii** is better than the first one.
