21921	DWDM
	Jutorial-5
	Ul8co021 - Suhil Bonde
g I	Converting decevior tree to rule is preferred because:
Service and the service and th	Converting to rules allows distinguishing among different contents in which a decesion node is used
Polygon de la companya de la company	It removes the distinction between attribute tests occurring at the res root & leaves of the tree
	It improves readability
No. of the control of	
A go desirence of the second	
An age of the second contract of the second c	



92	
a	To consider count of data tude, it
	needs to be order included in
.)	To consider count of data tuple, it seeds to be trobe included in the attribute selection measure
.3.	(such as information gain)
.5.	2000 20 10 20 10 10 10 10 10 10 10 10 10 10 10 10 10
6.	Solary Rayer: A = {26-30 kg, B= {31-35 kg
	C= {36-40k3, D= {41-45k3
	E = {46-50 k3, F= £66-7043
8.3	5.0 1 1 1 1 2
	{A,B, 03 (Sulary) {C, F3
0	
	{5}
	jowier Senior
	(Department)
€5	ecretary, system? { Sales, marketing?
	owor Sewor
	7
C	X = { Systems, 26 30, 46-50k}
	P(X journ) = 0.182
	P(x jouior) = 0.182 P(x serior) = 0
	C162 D D D D
	NB predicts janor

<i>Q</i> 3	
Tople	Class P. TP FP TTN EN TPR FPR
1	P 0.95 1 0 5 4 0.2 0
2	N 0.85 1 1 4 4 0.2 0.2
3	P 0.78 2 1 4 3 0.40-2
4	P 0.76 3 1 4 2 0.60-2
.5	N- 0.60 3 2 3 2 0.60.4
6	P 0.55 4 2 3 1 0.8 0.1
7	N 0.53 4 3 2 1 0.8 0.6
8	N 0.52 h h 1 1 08 0.8
9	N 0.51 4 5 0 1 0.8 1
10	P 040 5 5 8 0 1 0
	1 2 833
	Leines Senier
	D-8 (Nepatinent) 8-0
5.	
7.	6:6 m. of 6 & Super of the 63.
	0·h
	0.2
	. 531.0 : (
	0 0.2 0.4 0.6 0.8
	ROC Carre
	13 Madrille imir
-	
1	

04	Methode has class into laces.
	Methods for class imbalance:
	On the Part of the
	therampling: resample sometom tuples from
	Oversampling: Resample soundon tuples from numority class
	4
ii	Mudersampling: Delete random taples from
	Teamine of al
	majorety dass.
	7 10 11 (1)
icl	I hreshold Moving: Shifting the threshold
	to seduce the cartle
	Take neerting essen
-	Threshold Moving: Shifting the threshold to reduce the costly false negative errors.
	Cat Will the Al
_	Touthing . Indesing the Cost of
	mes-classifying the
	Cost-Modifying Incresing the cost of minority class
سرا	Eusenble mothods : Using boseting and
	The state of the s
	Ensemble methods: Using booting and random foreste.
	1
	5 VM & MLP would perform better on
	embalanced datasts than tree based
	5VM & MLP would perform better on imbalanced datasts than tree laser classifiers. SVM is not sensitive to imbalance as support vectors only look for boundries between clusters.
	into la con as convert yestore only lank
	moveme de suppero vectore and mor
	for voundries between cluster.
	Oversampling perfors better for numeric datasets like a credit and fraud transaction. Cost Modifying would help too.
	like a createt and hourd traverte
	CA 11 111
	case roomsying walla help too.

DWDM Tutorial 5

U18CO021: SAHIL BONDRE

5. Experiment with any dataset of your choice using a decision tree model, logistic regression, naïve Bayes classifier, and a support vector model. Make a table of your results. Are the differences in model performance significant?

1 Import Libraries

```
[1]: import numpy as np
  import pandas as pd
  from sklearn.datasets import load_iris
  from sklearn.model_selection import train_test_split
  from sklearn.preprocessing import StandardScaler
```

2 Load Data

```
[2]: # IRIS Dataset
X, y = load_iris(return_X_y=True)
print(X.shape, y.shape)

(150, 4) (150,)
```

3 Pre Processing

```
[3]: # No NAN Values
np.isnan(X).all() or np.isnan(y).all()
```

[3]: False

```
[4]:  # Splitting Test Train Dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
[5]: # Normalisation
scalar_X = StandardScaler()

X_train = scalar_X.fit_transform(X_train)
X_test = scalar_X.transform(X_test)
```

4 Training

```
[6]: from sklearn.tree import DecisionTreeClassifier
      from sklearn.linear_model import LogisticRegression
      from sklearn.naive_bayes import GaussianNB
      from sklearn.svm import SVC
      from sklearn.metrics import accuracy_score, f1_score
 [7]: dt = DecisionTreeClassifier()
      lr = LogisticRegression()
      nb = GaussianNB()
      svm = SVC()
 [8]: # Training
      dt.fit(X_train, y_train)
      lr.fit(X_train, y_train)
      nb.fit(X_train, y_train)
      svm.fit(X_train, y_train)
 [8]: SVC()
     5 Analysis
 [9]: y_pred_dt = dt.predict(X_test)
      y_pred_lr = lr.predict(X_test)
      y_pred_nb = nb.predict(X_test)
      y_pred_svm = svm.predict(X_test)
[10]: pred = [y_pred_dt, y_pred_lr, y_pred_nb, y_pred_svm]
      f1 = [f1_score(y_test, y, average="macro") for y in pred]
      acc = [accuracy_score(y_test, y) for y in pred]
[11]: print("\033[1m F1 Score \033[0m")
      classifier = ["Decesion Tree", "Logistic Regression", "Naive Bayes", "SVM"]
      for i in range(len(f1)):
          print(f"{classifier[i]}: {f1[i]:0.2f}")
      print()
      print("\033[1m Accuracy \033[0m")
      for i in range(len(f1)):
          print(f"{classifier[i]}: {acc[i]:0.2f}")
      F1 Score
     Decesion Tree: 0.97
```

Logistic Regression: 0.97

Naive Bayes: 0.97

SVM: 0.94

Accuracy

Decesion Tree: 0.97

Logistic Regression: 0.97

Naive Bayes: 0.97

SVM: 0.93