CSC 5741: Jupyter Notebook—Data Transformation Techniques

Lighton Phiri
<lighton.phiri@unza.zm>

June 7 2021

Contents

Introduction

In this Jupyter Notebook, perform the following basic data transformation tasks:

1. Transforming Categorical Attributes
   - Norminal Attributes
   - Ordinal Attributes
2. Bag-of-Words Model
3. Term Frequency
4. TF-IDF

NOTE: Derived attribtes from "Dataset #1: Initial Survey" and "Dataset #2: Student Demographics" have been included. Remember that this was meant to be an exercise. Ensure you download updated Jupyter Notebooks.

You will notice that the some examples use native Python features as opposed to libraries such as Pandas. This is done to highlight the flexibility that Python provides. In cases were they are not used, you are encouraged to explore how Pandas and other libraries can be used.

In all instances, you are encouraged to make reference to online documentation for the various tools. Additionally, you can exploit tools like Zeal Offline Documentation Browser to download and search through offline documentation. You are also encouraged to look up and explore other libraries, especially as you work towards the Mini Projects.

Recap on CRISP-DM Data Preparation Phase

A reminder about the following key activities conducted as part of data preparation:

- Derivation of new attributes from existing ones
- Scaling of attributes using appropriate ranges
- ***Data transformation into a form expected by the estimators***

General Notebook Configuration

[1]:
```
# Aesthetics for pandas cell outsput
import pandas as pd

pd.set_option('display.latex.repr', True)
pd.set_option('display.latex.longtable', True)
pd.set_option('max_colwidth', 30)

# Show all Jupyter Notebook cell output
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

Python Packages for Data Pre-processing

[2]:
```
# Import all libraries and modules for use during lecture session code walkthrough
import joblib
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import re
import seaborn as sns
import string

from collections import Counter
from IPython.core.interactiveshell import InteractiveShell
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import OrdinalEncoder
```

Load Save Pipelines

ICT 1110 Merged Dataset

- Remember that the merged dataset combines the three main datasets used in the analysis
  - Dataset #1: Initial Student Survey
  - Dataset #2: Student Demographics
  - Dataset #3: Student Assessment Scores

[3]:
```
# Load saved pipeline state
var_ict1110_student_performance = joblib.load("var_ict1110_student_performance.pkl")
```

[4]:
```
# Inspect loaded pipeline state
var_ict1110_student_performance.columns
#
#
len(var_ict1110_student_performance)
```

[4]:
```
Index(['Timestamp', 'StudentName', 'StudentID', 'HomeTown', 'MinorProgramme',
       'MinorProgrammeMotivation', 'MajorProgrammeMotivation',
       'DidComputerStudies', 'HasComputerTraining', 'ComputerTrainingType',
       ...
       'Test3Grade', 'Test3GPA', 'Test4Status', 'Test4Grade', 'Test4GPA',
       'ExaminationScore', 'ExaminationPCT', 'ExaminationStatus',
       'ExaminationGrade', 'ExaminationGPA'],
      dtype='object', length=153)
```

[4]:
```
123
```

[5]:
```
#
#
var_ict1110_student_performance.head(3).T
```

[5]:

	0	1	2
Timestamp	2019/03/28 11:13:51 PM GMT+2	2019/03/28 11:55:27 PM GMT+2	2019/(
StudentName	Participant1	Participant2	Partici
StudentID	NaN	742b8abe5776a6d942a92ce7dc...	921855
HomeTown	Chudleigh/Lusaka/Lusaka	Copperbelt,luanshya,Mpatamato	Mungu
MinorProgramme	Data Mining	Mathematics	Langu
MinorProgrammeMotivation	love data	find easi studi understand	best av
MajorProgrammeMotivation	love comput	want acquir knowledg ict c...	alway
DidComputerStudies	No	No	No
HasComputerTraining	Yes	No	No
ComputerTrainingType	I have studied Computer Sc...	MISSING DATA	MISSII
ExperienceWithComputers	More than 5 years	1 to 2 years	No Ex
HasComputerAccess	Yes	Yes	Yes
AboutMe	cycl everyday	day pass without joke feel...	
year	2019	2019	2019
StudentLocation	Lusaka	Copperbelt	Wester
DateOfBirth	NaN	1999-03-23	1999-1
Gender	NaN	F	M
AcademicYear	NaN	20191	20181
YearOfStudy	NaN	2nd Year	1st Yea
School	NaN	EDUCATION	EDUC
Program	NaN	BACHELOR OF INFORMATION AN...	BACH
MajorDescription	NaN	ICTs and Education	ICTs a
MinorDescription	NaN	MATHEMATICS	FRENC
Status	NaN	Registered	Regist
Sponsor	NaN	GRZ-FULLY SPONSORED	GRZ-F
Nationality	NaN	ZAMBIAN	ZAMB
Accommodated	NaN	No	Yes
StudentAge	NaN	20	19
Quiz1Score	NaN	5	5
Quiz2Score	NaN	4	4
Quiz3Score	NaN	2	6
Quiz4Score	NaN	3	4
Quiz5Score	NaN	6	9

	0	1	2
Quiz6Score	NaN	9	10
Quiz7Score	NaN	0	8
Quiz8Score	NaN	5	5
Quiz9Score	NaN	7	9.5
Quiz10Score	NaN	2	7
Quiz11Score	NaN	2	6
Quiz12Score	NaN	10	10
Quiz13Score	NaN	0	0.5
Quiz14Score	NaN	6	5
Quiz15Score	NaN	7	8
Quiz16Score	NaN	1	2
Quiz17Score	NaN	10	0
Quiz18Score	NaN	2	9
Quiz19Score	NaN	8	9
Quiz20Score	NaN	9	10
Quiz1PCT	NaN	50	50
Quiz2PCT	NaN	40	40
Quiz3PCT	NaN	20	60
Quiz4PCT	NaN	30	40
Quiz5PCT	NaN	60	90
Quiz6PCT	NaN	90	100
Quiz7PCT	NaN	0	80
Quiz8PCT	NaN	50	50
Quiz9PCT	NaN	70	95
Quiz10PCT	NaN	20	70
Quiz11PCT	NaN	20	60
Quiz12PCT	NaN	100	100
Quiz13PCT	NaN	0	5
Quiz14PCT	NaN	60	50
Quiz15PCT	NaN	70	80
Quiz16PCT	NaN	10	20
Quiz17PCT	NaN	100	0
Quiz18PCT	NaN	20	90
Quiz19PCT	NaN	80	90
Quiz20PCT	NaN	90	100
Quiz1Status	NaN	Pass	Pass
Quiz1Grade	NaN	C+	C+
Quiz1GPA	NaN	2.37	2.37
Quiz2Status	NaN	Fail	Fail
Quiz2Grade	NaN	D+	D+
Quiz2GPA	NaN	0	0
Quiz3Status	NaN	Fail	Pass
Quiz3Grade	NaN	D	B
Quiz3GPA	NaN	0	3
Quiz4Status	NaN	Fail	Fail
Quiz4Grade	NaN	D	D+
Quiz4GPA	NaN	0	0
Quiz5Status	NaN	Pass	Pass

	0	1	2
Quiz5Grade	NaN	B	A+
Quiz5GPA	NaN	3	5
Quiz6Status	NaN	Pass	Pass
Quiz6Grade	NaN	A+	A+
Quiz6GPA	NaN	5	5
Quiz7Status	NaN	Fail	Pass
Quiz7Grade	NaN	D	A
Quiz7GPA	NaN	0	4
Quiz8Status	NaN	Pass	Pass
Quiz8Grade	NaN	C+	C+
Quiz8GPA	NaN	2.37	2.37
Quiz9Status	NaN	Pass	Pass
Quiz9Grade	NaN	B+	A+
Quiz9GPA	NaN	3.5	5
Quiz10Status	NaN	Fail	Pass
Quiz10Grade	NaN	D	B+
Quiz10GPA	NaN	0	3.5
Quiz11Status	NaN	Fail	Pass
Quiz11Grade	NaN	D	B
Quiz11GPA	NaN	0	3
Quiz12Status	NaN	Pass	Pass
Quiz12Grade	NaN	A+	A+
Quiz12GPA	NaN	5	5
Quiz13Status	NaN	Fail	Fail
Quiz13Grade	NaN	D	D
Quiz13GPA	NaN	0	0
Quiz14Status	NaN	Pass	Pass
Quiz14Grade	NaN	B	C+
Quiz14GPA	NaN	3	2.37
Quiz15Status	NaN	Pass	Pass
Quiz15Grade	NaN	B+	A
Quiz15GPA	NaN	3.5	4
Quiz16Status	NaN	Fail	Fail
Quiz16Grade	NaN	D	D
Quiz16GPA	NaN	0	0
Quiz17Status	NaN	Pass	Fail
Quiz17Grade	NaN	A+	D
Quiz17GPA	NaN	5	0
Quiz18Status	NaN	Fail	Pass
Quiz18Grade	NaN	D	A+
Quiz18GPA	NaN	0	5
Quiz19Status	NaN	Pass	Pass
Quiz19Grade	NaN	A	A+
Quiz19GPA	NaN	4	5
Quiz20Status	NaN	Pass	Pass
Quiz20Grade	NaN	A+	A+
Quiz20GPA	NaN	5	5
Test1Score	NaN	24.5	40

	0	1	2
Test2Score	NaN	15.5	16
Test3Score	NaN	21	24
Test4Score	NaN	17.5	22
Test1PCT	NaN	49	80
Test2PCT	NaN	31	32
Test3PCT	NaN	42	48
Test4PCT	NaN	35	44
Test1Status	NaN	Pass	Pass
Test1Grade	NaN	C	A
Test1GPA	NaN	1.5	4
Test2Status	NaN	Fail	Fail
Test2Grade	NaN	D	D
Test2GPA	NaN	0	0
Test3Status	NaN	Fail	Pass
Test3Grade	NaN	D+	C
Test3GPA	NaN	0	1.5
Test4Status	NaN	Fail	Fail
Test4Grade	NaN	D	D+
Test4GPA	NaN	0	0
ExaminationScore	NaN	56	85
ExaminationPCT	NaN	56	85
ExaminationStatus	NaN	Pass	Pass
ExaminationGrade	NaN	C+	A
ExaminationGPA	NaN	2.37	4

[6]:
```
#
# Filter records where students wrote the final examination
# Note: Analysis will be based on observations where students wrote the final␣
↪examination
#
var_ict1110_student_performance_dataframe =␣
↪var_ict1110_student_performance[var_ict1110_student_performance["ExaminationScore"].
↪notna()]
```

[7]:
```
#
# Inspect filtered observations
#
var_ict1110_student_performance_dataframe.head(3).T
len(var_ict1110_student_performance_dataframe)
```

[7]:

	1	2
Timestamp	2019/03/28 11:55:27 PM GMT+2	2019/03/29 8:00:53 PM GMT+2
StudentName	Participant2	Participant3
StudentID	742b8abe5776a6d942a92ce7dc...	921855f753932de762b780405a...
HomeTown	Copperbelt,luanshya,Mpatamato	Mungule,senanga,western.
MinorProgramme	Mathematics	Languages
MinorProgrammeMotivation	find easi studi understand	best avail option

	1	2
MajorProgrammeMotivation	want acquir knowledg ict c...	alway want ict relat program
DidComputerStudies	No	No
HasComputerTraining	No	No
ComputerTrainingType	MISSING DATA	MISSING DATA
ExperienceWithComputers	1 to 2 years	No Experience
HasComputerAccess	Yes	Yes
AboutMe	day pass without joke feel...	
year	2019	2019
StudentLocation	Copperbelt	Western
DateOfBirth	1999-03-23	1999-12-24
Gender	F	M
AcademicYear	20191	20181
YearOfStudy	2nd Year	1st Year
School	EDUCATION	EDUCATION
Program	BACHELOR OF INFORMATION AN...	BACHELOR OF INFORMATION AN...
MajorDescription	ICTs and Education	ICTs and Education
MinorDescription	MATHEMATICS	FRENCH
Status	Registered	Registered
Sponsor	GRZ-FULLY SPONSORED	GRZ-FULLY SPONSORED
Nationality	ZAMBIAN	ZAMBIAN
Accommodated	No	Yes
StudentAge	20	19
Quiz1Score	5	5
Quiz2Score	4	4
Quiz3Score	2	6
Quiz4Score	3	4
Quiz5Score	6	9
Quiz6Score	9	10
Quiz7Score	0	8
Quiz8Score	5	5
Quiz9Score	7	9.5
Quiz10Score	2	7
Quiz11Score	2	6
Quiz12Score	10	10
Quiz13Score	0	0.5
Quiz14Score	6	5
Quiz15Score	7	8
Quiz16Score	1	2
Quiz17Score	10	0
Quiz18Score	2	9
Quiz19Score	8	9
Quiz20Score	9	10
Quiz1PCT	50	50
Quiz2PCT	40	40
Quiz3PCT	20	60
Quiz4PCT	30	40
Quiz5PCT	60	90
Quiz6PCT	90	100

	1	2
Quiz7PCT	0	80
Quiz8PCT	50	50
Quiz9PCT	70	95
Quiz10PCT	20	70
Quiz11PCT	20	60
Quiz12PCT	100	100
Quiz13PCT	0	5
Quiz14PCT	60	50
Quiz15PCT	70	80
Quiz16PCT	10	20
Quiz17PCT	100	0
Quiz18PCT	20	90
Quiz19PCT	80	90
Quiz20PCT	90	100
Quiz1Status	Pass	Pass
Quiz1Grade	C+	C+
Quiz1GPA	2.37	2.37
Quiz2Status	Fail	Fail
Quiz2Grade	D+	D+
Quiz2GPA	0	0
Quiz3Status	Fail	Pass
Quiz3Grade	D	B
Quiz3GPA	0	3
Quiz4Status	Fail	Fail
Quiz4Grade	D	D+
Quiz4GPA	0	0
Quiz5Status	Pass	Pass
Quiz5Grade	B	A+
Quiz5GPA	3	5
Quiz6Status	Pass	Pass
Quiz6Grade	A+	A+
Quiz6GPA	5	5
Quiz7Status	Fail	Pass
Quiz7Grade	D	A
Quiz7GPA	0	4
Quiz8Status	Pass	Pass
Quiz8Grade	C+	C+
Quiz8GPA	2.37	2.37
Quiz9Status	Pass	Pass
Quiz9Grade	B+	A+
Quiz9GPA	3.5	5
Quiz10Status	Fail	Pass
Quiz10Grade	D	B+
Quiz10GPA	0	3.5
Quiz11Status	Fail	Pass
Quiz11Grade	D	B
Quiz11GPA	0	3
Quiz12Status	Pass	Pass

	1	2
Quiz12Grade	A+	A+
Quiz12GPA	5	5
Quiz13Status	Fail	Fail
Quiz13Grade	D	D
Quiz13GPA	0	0
Quiz14Status	Pass	Pass
Quiz14Grade	B	C+
Quiz14GPA	3	2.37
Quiz15Status	Pass	Pass
Quiz15Grade	B+	A
Quiz15GPA	3.5	4
Quiz16Status	Fail	Fail
Quiz16Grade	D	D
Quiz16GPA	0	0
Quiz17Status	Pass	Fail
Quiz17Grade	A+	D
Quiz17GPA	5	0
Quiz18Status	Fail	Pass
Quiz18Grade	D	A+
Quiz18GPA	0	5
Quiz19Status	Pass	Pass
Quiz19Grade	A	A+
Quiz19GPA	4	5
Quiz20Status	Pass	Pass
Quiz20Grade	A+	A+
Quiz20GPA	5	5
Test1Score	24.5	40
Test2Score	15.5	16
Test3Score	21	24
Test4Score	17.5	22
Test1PCT	49	80
Test2PCT	31	32
Test3PCT	42	48
Test4PCT	35	44
Test1Status	Pass	Pass
Test1Grade	C	A
Test1GPA	1.5	4
Test2Status	Fail	Fail
Test2Grade	D	D
Test2GPA	0	0
Test3Status	Fail	Pass
Test3Grade	D+	C
Test3GPA	0	1.5
Test4Status	Fail	Fail
Test4Grade	D	D+
Test4GPA	0	0
ExaminationScore	56	85
ExaminationPCT	56	85

	1	2
ExaminationStatus	Pass	Pass
ExaminationGrade	C+	A
ExaminationGPA	2.37	4

[7]: 63

Data Attribute Selection

Before model implementation, you want to ensure you select relevant attributes to use as independent variables

In the case of the ICT 1110 example we end up working with the following attributes: * Initial Survey Dataset * StudentID * HomeTown * MinorProgrammeMotivation * MajorProgrammeMotivation * DidComputerStudies * HasComputerTraining * ExperienceWithComputers * HasComputerAccess * Student Demographics Dataset * StudentAge (derived from DateOfBirth) * Gender * MinorDescription * Sponsor * Accommodated * Assessment Scores Dataset * XPCT * XStatus * XGrade * XGPA

[8]:
```
#
# Create a new DataFrame with desired columns
#
var_initial_survey_columns = ["StudentID", "StudentLocation",␣
↪"MinorProgrammeMotivation", "MajorProgrammeMotivation", "DidComputerStudies",␣
↪"HasComputerTraining", "ExperienceWithComputers", "HasComputerAccess"]
#
#
var_student_demographics_columns = ["StudentAge", "Gender", "MinorDescription",␣
↪"Sponsor", "Accommodated"]
#
#
var_assessment_scores_columns = [
    var_column for var_column in var_ict1110_student_performance_dataframe.columns if (
        var_column.endswith("PCT") or
        var_column.endswith("Status") or
        var_column.endswith("Grade") or
        var_column.endswith("GPA")
    )
]
#
# Concatenate desired columns
#
var_ict1110_student_performance_dataframe_columns = var_initial_survey_columns +␣
↪var_student_demographics_columns + var_assessment_scores_columns

print (var_ict1110_student_performance_dataframe_columns)
len(var_ict1110_student_performance_dataframe_columns)
#
#
#
```

```
var_ict1110_student_performance_dataframe_selected =␣
↪var_ict1110_student_performance_dataframe[var_ict1110_student_performance_dataframe_columns]
```

['StudentID', 'StudentLocation', 'MinorProgrammeMotivation', 'MajorProgrammeMotivation', 'DidComputerStudies', 'HasComputerTraining', 'ExperienceWithComputers', 'HasComputerAccess', 'StudentAge', 'Gender', 'MinorDescription', 'Sponsor', 'Accommodated', 'Status', 'Quiz1PCT', 'Quiz2PCT', 'Quiz3PCT', 'Quiz4PCT', 'Quiz5PCT', 'Quiz6PCT', 'Quiz7PCT', 'Quiz8PCT', 'Quiz9PCT', 'Quiz10PCT', 'Quiz11PCT', 'Quiz12PCT', 'Quiz13PCT', 'Quiz14PCT', 'Quiz15PCT', 'Quiz16PCT', 'Quiz17PCT', 'Quiz18PCT', 'Quiz19PCT', 'Quiz20PCT', 'Quiz1Status', 'Quiz1Grade', 'Quiz1GPA', 'Quiz2Status', 'Quiz2Grade', 'Quiz2GPA', 'Quiz3Status', 'Quiz3Grade', 'Quiz3GPA', 'Quiz4Status', 'Quiz4Grade', 'Quiz4GPA', 'Quiz5Status', 'Quiz5Grade', 'Quiz5GPA', 'Quiz6Status', 'Quiz6Grade', 'Quiz6GPA', 'Quiz7Status', 'Quiz7Grade', 'Quiz7GPA', 'Quiz8Status', 'Quiz8Grade', 'Quiz8GPA', 'Quiz9Status', 'Quiz9Grade', 'Quiz9GPA', 'Quiz10Status', 'Quiz10Grade', 'Quiz10GPA', 'Quiz11Status', 'Quiz11Grade', 'Quiz11GPA', 'Quiz12Status', 'Quiz12Grade', 'Quiz12GPA', 'Quiz13Status', 'Quiz13Grade', 'Quiz13GPA', 'Quiz14Status', 'Quiz14Grade', 'Quiz14GPA', 'Quiz15Status', 'Quiz15Grade', 'Quiz15GPA', 'Quiz16Status', 'Quiz16Grade', 'Quiz16GPA', 'Quiz17Status', 'Quiz17Grade', 'Quiz17GPA', 'Quiz18Status', 'Quiz18Grade', 'Quiz18GPA', 'Quiz19Status', 'Quiz19Grade', 'Quiz19GPA', 'Quiz20Status', 'Quiz20Grade', 'Quiz20GPA', 'Test1PCT', 'Test2PCT', 'Test3PCT', 'Test4PCT', 'Test1Status', 'Test1Grade', 'Test1GPA', 'Test2Status', 'Test2Grade', 'Test2GPA', 'Test3Status', 'Test3Grade', 'Test3GPA', 'Test4Status', 'Test4Grade', 'Test4GPA', 'ExaminationPCT', 'ExaminationStatus', 'ExaminationGrade', 'ExaminationGPA']

[8]: 114

[9]:
```
#
# Inspect DataFrame with selected columns
#
var_ict1110_student_performance_dataframe_selected.head(3).T
```

[9]:

	1	2	3
StudentID	742b8abe5776a6d942a92ce7dc...	921855f753932de762b780405a...	07f3ca235faaa1c
StudentLocation	Copperbelt	Western	Lusaka
MinorProgrammeMotivation	find easi studi understand	best avail option	chose
MajorProgrammeMotivation	want acquir knowledg ict c...	alway want ict relat program	result met requi
DidComputerStudies	No	No	No
HasComputerTraining	No	No	No
ExperienceWithComputers	1 to 2 years	No Experience	Less than 1 year
HasComputerAccess	Yes	Yes	Yes
StudentAge	20	19	22
Gender	F	M	M
MinorDescription	MATHEMATICS	FRENCH	RELIGIOUS STU
Sponsor	GRZ-FULLY SPONSORED	GRZ-FULLY SPONSORED	GRZ-FULLY SPO
Accommodated	No	Yes	No
Status	Registered	Registered	Registered
Quiz1PCT	50	50	60
Quiz2PCT	40	40	40
Quiz3PCT	20	60	60

Continued

	1	2	3
Quiz4PCT	30	40	50
Quiz5PCT	60	90	20
Quiz6PCT	90	100	90
Quiz7PCT	0	80	10
Quiz8PCT	50	50	60
Quiz9PCT	70	95	75
Quiz10PCT	20	70	10
Quiz11PCT	20	60	50
Quiz12PCT	100	100	90
Quiz13PCT	0	5	40
Quiz14PCT	60	50	80
Quiz15PCT	70	80	80
Quiz16PCT	10	20	100
Quiz17PCT	100	0	40
Quiz18PCT	20	90	90
Quiz19PCT	80	90	100
Quiz20PCT	90	100	100
Quiz1Status	Pass	Pass	Pass
Quiz1Grade	C+	C+	B
Quiz1GPA	2.37	2.37	3
Quiz2Status	Fail	Fail	Fail
Quiz2Grade	D+	D+	D+
Quiz2GPA	0	0	0
Quiz3Status	Fail	Pass	Pass
Quiz3Grade	D	B	B
Quiz3GPA	0	3	3
Quiz4Status	Fail	Fail	Pass
Quiz4Grade	D	D+	C+
Quiz4GPA	0	0	2.37
Quiz5Status	Pass	Pass	Fail
Quiz5Grade	B	A+	D
Quiz5GPA	3	5	0
Quiz6Status	Pass	Pass	Pass
Quiz6Grade	A+	A+	A+
Quiz6GPA	5	5	5
Quiz7Status	Fail	Pass	Fail
Quiz7Grade	D	A	D
Quiz7GPA	0	4	0
Quiz8Status	Pass	Pass	Pass
Quiz8Grade	C+	C+	B
Quiz8GPA	2.37	2.37	3
Quiz9Status	Pass	Pass	Pass
Quiz9Grade	B+	A+	B+
Quiz9GPA	3.5	5	3.5
Quiz10Status	Fail	Pass	Fail
Quiz10Grade	D	B+	D
Quiz10GPA	0	3.5	0
Quiz11Status	Fail	Pass	Pass

Continued

	1	2	3
Quiz11Grade	D	B	C+
Quiz11GPA	0	3	2.37
Quiz12Status	Pass	Pass	Pass
Quiz12Grade	A+	A+	A+
Quiz12GPA	5	5	5
Quiz13Status	Fail	Fail	Fail
Quiz13Grade	D	D	D+
Quiz13GPA	0	0	0
Quiz14Status	Pass	Pass	Pass
Quiz14Grade	B	C+	A
Quiz14GPA	3	2.37	4
Quiz15Status	Pass	Pass	Pass
Quiz15Grade	B+	A	A
Quiz15GPA	3.5	4	4
Quiz16Status	Fail	Fail	Pass
Quiz16Grade	D	D	A+
Quiz16GPA	0	0	5
Quiz17Status	Pass	Fail	Fail
Quiz17Grade	A+	D	D+
Quiz17GPA	5	0	0
Quiz18Status	Fail	Pass	Pass
Quiz18Grade	D	A+	A+
Quiz18GPA	0	5	5
Quiz19Status	Pass	Pass	Pass
Quiz19Grade	A	A+	A+
Quiz19GPA	4	5	5
Quiz20Status	Pass	Pass	Pass
Quiz20Grade	A+	A+	A+
Quiz20GPA	5	5	5
Test1PCT	49	80	63
Test2PCT	31	32	59
Test3PCT	42	48	68
Test4PCT	35	44	59
Test1Status	Pass	Pass	Pass
Test1Grade	C	A	B
Test1GPA	1.5	4	3
Test2Status	Fail	Fail	Pass
Test2Grade	D	D	C+
Test2GPA	0	0	2.37
Test3Status	Fail	Pass	Pass
Test3Grade	D+	C	B
Test3GPA	0	1.5	3
Test4Status	Fail	Fail	Pass
Test4Grade	D	D+	C+
Test4GPA	0	0	2.37
ExaminationPCT	56	85	64
ExaminationStatus	Pass	Pass	Pass
ExaminationGrade	C+	A	B

Continued

	1	2	3
ExaminationGPA	2.37	4	3

Data Transformation Techniques

Most estimators expect that the input data attributes associated with observations are numeric. With that in mind, all data attributes to be used as input to the estimators must be converted into numeric form.

In the case of the ICT 1110 dataset, the attribute types are as follows: * Initial Survey Dataset * StudentID (N/A—Field used as unique identifier) * HomeTown—Categorical to One-Hot-Encoding * MinorProgrammeMotivation—Text to TF/TF-IDF * MajorProgrammeMotivation—Text to TF/TF-IDF * DidComputerStudies—Categorical to One-Hot-Encoding * HasComputerTraining—Categorical to One-Hot-Encoding * ExperienceWithComputers—Ordinal to Numeric Encoding * HasComputerAccess—Categorical to On-Hot-Encoding * Student Demographics Dataset * StudentAge—Numeric to Unchanged * Gender—Categorical to One-Hot-Encoding * MinorDescription—Categorical to One-Hot-Encoding * Sponsor—Categorical to One-Hot-Encoding * Accommodated—Categorical to One-Hot-Encoding * Assessment Scores Dataset * XPCT—Numeric to Unchanged * XStatus—Categorical to One-Hot-Encoding * XGrade—Categorical to One-Hot-Encoding * XGPA—Categorical to One-Hot-Encoding

Encoding Labels

We will work with a sample outcome label attribute with with more than two possible values * ExaminationGrade—A+, A, B+, B, C+, C, D+ and D

Assuming a hypothetical model that aims to ascertain the influence of Minors on final examination pass/fail status

[10]:
```
#
# Create DataFrame for experimentation
#
var_ict1110_student_performance_label_example␣
 ↪=var_ict1110_student_performance_dataframe_selected[var_ict1110_student_performance_dataframe_s
 ↪notna()][["StudentID", "ExperienceWithComputers", "ExaminationGrade"]]

#
#
var_ict1110_student_performance_label_example.head(3)
```

[10]:

	StudentID	ExperienceWithComputers	ExaminationGrade
1	742b8abe5776a6d942a92ce7dc...	1 to 2 years	C+
2	921855f753932de762b780405a...	No Experience	A
3	07f3ca235faaa1c9ad16facef5...	Less than 1 year	B

[11]:
```
#
# Inspect unique entries associated with ExperienceWithComputers
#
var_ict1110_student_performance_label_example["ExaminationGrade"].unique()
```

[11]:
```
array(['C+', 'A', 'B', 'D', 'C', 'D+', 'B+'], dtype=object)
```

[12]:
```
#
# Variable for dictionary example approach
var_ict1110_student_performance_label_example1 =␣
↪var_ict1110_student_performance_label_example
```

[13]:
```
#
# Use OrdinalEncoder to transform the entries
# from sklearn.preprocessing import LabelEncoder
# categories parameter provides a mechanism to specify natural order of categories
#
var_examination_grade_le = LabelEncoder()
```

[14]:
```
#
# Inspect the categories
var_examination_grade_le.
↪fit_transform(var_ict1110_student_performance_label_example["ExaminationGrade"])
```

[14]:
```
array([4, 0, 1, 1, 4, 4, 4, 1, 4, 5, 5, 3, 0, 1, 1, 6, 6, 6, 5, 5, 6, 6,
       6, 5, 0, 3, 1, 2, 1, 4, 5, 1, 1, 0, 4, 3, 4, 4, 1, 6, 5, 5, 6, 4,
       1, 4, 3, 4, 1, 4, 6, 4, 5, 3, 4, 1, 6, 5, 2, 2, 5, 2, 1])
```

[15]:
```
var_examination_grade_le.classes_
```

[15]:
```
array(['A', 'B', 'B+', 'C', 'C+', 'D', 'D+'], dtype=object)
```

[16]:
```
#
# Create new DataFrame column with encoded labels
#
var_ict1110_student_performance_label_example1["ExaminationGradeEncoder"] =␣
↪var_examination_grade_le.
↪fit_transform(var_ict1110_student_performance_label_example1["ExaminationGrade"])
```

[17]:
```
var_ict1110_student_performance_label_example1.head(5)
```

[17]:

	StudentID	ExperienceWithComputers	ExaminationGrade	ExaminationGradeEncode
1	742b8abe5776a6d942a92ce7dc...	1 to 2 years	C+	
2	921855f753932de762b780405a...	No Experience	A	
3	07f3ca235faaa1c9ad16facef5...	Less than 1 year	B	
4	4234d1794dd33c1b6ed975eab5...	Less than 1 year	B	
5	9e7002d53d4db7bfad4f5cf419...	No Experience	C+	

Encoding Ordinal Attributes

We will work with a sample ordinal attribute with more than two possible values * ExperienceWithComputers * No Experience * Less than 1 year * 1 to 2 years * More than 5 years

Assuming a hypothetical model that aims to ascertain the influence of experience working with computers on final examination pass/fail status

[18]:
```
#
# Create DataFrame for experimentation
```

```
#
var_ict1110_student_performance_ordinal_example␣
↪=var_ict1110_student_performance_dataframe_selected[var_ict1110_student_performance_dataframe_s
↪notna()][["StudentID", "ExperienceWithComputers", "ExaminationStatus"]]
```

[19]:
```
#
# Inspect entries
var_ict1110_student_performance_ordinal_example.head(8)
```

[19]:

	StudentID	ExperienceWithComputers	ExaminationStatus
1	742b8abe5776a6d942a92ce7dc...	1 to 2 years	Pass
2	921855f753932de762b780405a...	No Experience	Pass
3	07f3ca235faaa1c9ad16facef5...	Less than 1 year	Pass
4	4234d1794dd33c1b6ed975eab5...	Less than 1 year	Pass
5	9e7002d53d4db7bfad4f5cf419...	No Experience	Pass
6	fceb5af40df295d85851f390f4...	Less than 1 year	Pass
7	e299d7cb0f7866cce7d90da2af...	More than 5 years	Pass
8	74458a3d3e5f3074226b1f9fa2...	No Experience	Pass

[20]:
```
#
# Inspect unique entries associated with ExperienceWithComputers
#
var_ict1110_student_performance_ordinal_example["ExperienceWithComputers"].unique()
```

[20]:
```
array(['1 to 2 years', 'No Experience', 'Less than 1 year',
       'More than 5 years', 'MISSING DATA'], dtype=object)
```

Using Dictionary Mapping

[21]:
```
#
# Variable for dictionary example approach
var_ict1110_student_performance_ordinal_example1 =␣
↪var_ict1110_student_performance_ordinal_example
```

[22]:
```
#
# Create dictionary mapping of ordinal values
var_computing_experience_dictionary = {
    'No Experience': 0,
    'Less than 1 year': 1,
    '1 to 2 years': 2,
    'More than 5 years': 3,
    'MISSING': 4
}
```

[23]:
```
var_ict1110_student_performance_ordinal_example1["ExperienceWithComputers"].
↪map(var_computing_experience_dictionary).head(2)
```

[23]:

	ExperienceWithComputers
1	2.0
2	0.0

[24]:
```
#
# Create new DataFrame column with encoded values
var_ict1110_student_performance_ordinal_example1["ExperienceWithComputersEncoded"] =␣
↪var_ict1110_student_performance_ordinal_example1["ExperienceWithComputers"].
↪map(var_computing_experience_dictionary)
```

[25]:
```
#
# Inspect encodings
var_ict1110_student_performance_ordinal_example1[["StudentID",␣
↪"ExperienceWithComputers", "ExperienceWithComputersEncoded"]].head(12)
```

[25]:

	StudentID	ExperienceWithComputers	ExperienceWithComputersEncoded
1	742b8abe5776a6d942a92ce7dc...	1 to 2 years	2.0
2	921855f753932de762b780405a...	No Experience	0.0
3	07f3ca235faaa1c9ad16facef5...	Less than 1 year	1.0
4	4234d1794dd33c1b6ed975eab5...	Less than 1 year	1.0
5	9e7002d53d4db7bfad4f5cf419...	No Experience	0.0
6	fceb5af40df295d85851f390f4...	Less than 1 year	1.0
7	e299d7cb0f7866cce7d90da2af...	More than 5 years	3.0
8	74458a3d3e5f3074226b1f9fa2...	No Experience	0.0
9	81975d05c61d8de83f46487739...	More than 5 years	3.0
10	2d65f5236205dd23c6a8212627...	No Experience	0.0
11	e03653dedd6f4e142f4aca1319...	1 to 2 years	2.0
12	e7400496f1ce70cb62c2c44ca2...	More than 5 years	3.0

Using OrdinalEncoder

[26]:
```
#
# Variable for dictionary example approach
var_ict1110_student_performance_ordinal_example2 =␣
↪var_ict1110_student_performance_ordinal_example
```

[27]:
```
var_ict1110_student_performance_ordinal_example2["ExperienceWithComputers"].unique()
```

[27]:
```
array(['1 to 2 years', 'No Experience', 'Less than 1 year',
       'More than 5 years', 'MISSING DATA'], dtype=object)
```

[28]:
```
#
# Use OrdinalEncoder to transform the entries
# from sklearn.preprocessing import OrdinalEncoder
# categories parameter provides a mechanism to specify natural order of categories
#
var_computing_experience_oe = OrdinalEncoder()
```

[29]:
```
#
# Inspect the categories
var_computing_experience_oe.categories
```

[29]: 'auto'

[30]:
```
#
# Fit and Transform the ordinal attributes
var_ict1110_student_performance_ordinal_example2["ExperienceWithComputersEncoded"] =␣
↪var_computing_experience_oe.
↪fit_transform(var_ict1110_student_performance_ordinal_example2[["ExperienceWithComputers"]])
#####var_computing_experience_oe?
```

[31]:
```
var_ict1110_student_performance_ordinal_example2[["StudentID",␣
↪"ExperienceWithComputers", "ExperienceWithComputersEncoded"]].head(8)
```

[31]:

	StudentID	ExperienceWithComputers	ExperienceWithComputersEncoded
1	742b8abe5776a6d942a92ce7dc...	1 to 2 years	0.0
2	921855f753932de762b780405a...	No Experience	4.0
3	07f3ca235faaa1c9ad16facef5...	Less than 1 year	1.0
4	4234d1794dd33c1b6ed975eab5...	Less than 1 year	1.0
5	9e7002d53d4db7bfad4f5cf419...	No Experience	4.0
6	fceb5af40df295d85851f390f4...	Less than 1 year	1.0
7	e299d7cb0f7866cce7d90da2af...	More than 5 years	3.0
8	74458a3d3e5f3074226b1f9fa2...	No Experience	4.0

Encoding Nominal Attributes

We will work with a sample nominal attribute with two possible values and another one with more than two possible values * ExaminationStatus—Fail/Pass * MinorDescription—Multiple Values

Assuming a hypothetical model that aims to ascertain the influence of Minors on final examination pass/fail status

[32]:
```
var_ict1110_student_performance_dataframe_selected[["StudentID", "MinorDescription",␣
↪"ExaminationStatus"]].tail(8)
```

[32]:

	StudentID	MinorDescription	ExaminationStatus
113	575b9408b6daa2ddcefbcf6d81...	ENGLISH	Pass
114	9c05c40b81fd906b1585e231c0...	MISSING DATA	Fail
115	aa293d284f52b08da5ba7fe779...	MISSING DATA	Fail
116	0a6355f86240762961997e70c5...	ENGLISH	Pass
117	0a6355f86240762961997e70c5...	ENGLISH	Pass
118	1e746c5c85c4c0de3a0858b0dd...	CIVIC EDUCATION	Fail
119	b0aa0804e676a38255af4fd702...	MATHEMATICS	Pass
120	c03b1123e45fa00da3142e0424...	ENGLISH	Pass

Using Pandas

- Pandas has a get_dummies function that can be used to one-hot-encode values

[33]:
```
var_ict1110_student_performance_dataframe_selected["ExaminationStatus"].head(10)
```

[33]:

	ExaminationStatus
1	Pass
2	Pass
3	Pass
4	Pass
5	Pass
6	Pass
7	Pass
8	Pass
9	Pass
10	Fail

[34]:
```
#
# Examination Status Attribute Values
#
var_ict1110_student_performance_dataframe_selected["ExaminationStatus"].unique()
pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_selected["ExaminationStatus"]).
 ↪head(10)
```

[34]:
```
array(['Pass', 'Fail'], dtype=object)
```

[34]:

	Fail	Pass
1	0	1
2	0	1
3	0	1
4	0	1
5	0	1
6	0	1
7	0	1
8	0	1
9	0	1
10	1	0

[35]:
```
var_ict1110_student_performance_dataframe_selected["MinorDescription"].head(3)
```

[35]:

	MinorDescription
1	MATHEMATICS
2	FRENCH
3	RELIGIOUS STUDIES

[36]:
```
#
# MinorDescription Attribute Values
#
```

```python
var_ict1110_student_performance_dataframe_selected["MinorDescription"].unique()
pd.get_dummies(var_ict1110_student_performance_dataframe_selected["MinorDescription"]).
↪head(3)
```

[36]:
```
array(['MATHEMATICS', 'FRENCH', 'RELIGIOUS STUDIES', 'CIVIC EDUCATION',
       'MISSING DATA', 'ENGLISH', 'HISTORY', 'ART AND DESIGN STUDIES',
       'GEOGRAPHY'], dtype=object)
```

[36]:

	ART AND DESIGN STUDIES	CIVIC EDUCATION	ENGLISH	FRENCH	GEOGRAPHY	HISTORY	M
1	0	0	0	0	0	0	
2	0	0	0	1	0	0	
3	0	0	0	0	0	0	

[37]:
```python
#
# Merge encoded values with main DataFrame
#
var_ict1110_student_performance_dataframe_categorical = pd.concat(
    [
        var_ict1110_student_performance_dataframe_selected[["StudentID",␣
↪"MinorDescription"]],
        pd.
↪get_dummies(var_ict1110_student_performance_dataframe_selected["MinorDescription"]),
        var_ict1110_student_performance_dataframe_selected["ExaminationStatus"],
        pd.
↪get_dummies(var_ict1110_student_performance_dataframe_selected["ExaminationStatus"])
    ],
    axis="columns"
)
```

[38]:
```python
var_ict1110_student_performance_dataframe_categorical.tail(8)
```

[38]:

	StudentID	MinorDescription	ART AND DESIGN STUDIES	CIVIC EDUCATIO
113	575b9408b6daa2ddcefbcf6d81...	ENGLISH	0	0
114	9c05c40b81fd906b1585e231c0...	MISSING DATA	0	0
115	aa293d284f52b08da5ba7fe779...	MISSING DATA	0	0
116	0a6355f86240762961997e70c5...	ENGLISH	0	0
117	0a6355f86240762961997e70c5...	ENGLISH	0	0
118	1e746c5c85c4c0de3a0858b0dd...	CIVIC EDUCATION	0	1
119	b0aa0804e676a38255af4fd702...	MATHEMATICS	0	0
120	c03b1123e45fa00da3142e0424...	ENGLISH	0	0

Using scikit-learn

Exercise: Rearch on how to use scikit-learn to OneHotEncode

Additional Encoding Techniques

Multiple Labels Using MultiLabelBinarizer

[39]:
```
#
# Assuming one want to classify ETD subjects/topics
#
from sklearn.preprocessing import MultiLabelBinarizer
var_etd_subjects = [
    ('Computer Science', 'Data Mining'),
    ('Education', 'Adult Education'),
    ('Computer Science', 'Software Engineering'),
    ('Engineering', 'Power Electrical'),
    ('Education', 'Languages')
]
```

[40]:
```
#
# MultiLabelBinarizer object
var_etd_subjects_mlb = MultiLabelBinarizer()

#
# Encode
var_etd_subjects_mlb.fit_transform(var_etd_subjects)

#
#
var_etd_subjects_mlb.classes_
```

[40]:
```
array([[0, 1, 1, 0, 0, 0, 0, 0],
       [1, 0, 0, 1, 0, 0, 0, 0],
       [0, 1, 0, 0, 0, 0, 0, 1],
       [0, 0, 0, 0, 1, 0, 1, 0],
       [0, 0, 0, 1, 0, 1, 0, 0]])
```

[40]:
```
array(['Adult Education', 'Computer Science', 'Data Mining', 'Education',
       'Engineering', 'Languages', 'Power Electrical',
       'Software Engineering'], dtype=object)
```

Exercise: Research DictVectorizer

Text Encoding

We will work with a sample text attribute from Dataset #1: Initial Survey * MajorProgrammeMotivation—This attribute has subject responses associated with the motivation for why students chose to pursue the programme

Assuming a hypothetical model that aims to ascertain the influence of motivation to pursue the course on final examination pass/fail status

On The Importance of Text Pre-Processing

Input text features require thorough cleaning to ensure that only relevant features are incorporated into the model. At a minimum, all common text pre-processing operations must be performed: * Case normalisation * Punctuation removal * Stopwords removal * Stemming

In certain instances, additional operations such as "Lemmatization", out-of-vocabulary words normalisation and noise removal.

In the case of the ICT 1110 dataset, basic text pre-processing operations have already been performed. Please see the Data Pre-Processing Lecture Series and corresponding Jupyter Notebook

[41]:
```
#
# Create DataFrame for experimentation
#
var_ict1110_student_performance_text_example␣
↪=var_ict1110_student_performance_dataframe_selected[var_ict1110_student_performance_dataframe_s
↪notna()][["StudentID", "MajorProgrammeMotivation", "ExaminationStatus"]]
```

[42]:
```
#
#
#
var_ict1110_student_performance_text_example.head(12)
len(var_ict1110_student_performance_text_example)
```

[42]:

	StudentID	MajorProgrammeMotivation	ExaminationStatus
1	742b8abe5776a6d942a92ce7dc...	want acquir knowledg ict c...	Pass
2	921855f753932de762b780405a...	alway want ict relat program	Pass
3	07f3ca235faaa1c9ad16facef5...	result met requir	Pass
4	4234d1794dd33c1b6ed975eab5...	written program twice appl...	Pass
5	9e7002d53d4db7bfad4f5cf419...	part parcel ever chang dev...	Pass
6	fceb5af40df295d85851f390f4...	alway want studi program	Pass
7	e299d7cb0f7866cce7d90da2af...	use comput lot	Pass
8	74458a3d3e5f3074226b1f9fa2...	know	Pass
9	81975d05c61d8de83f46487739...	know behind digit world	Pass
10	2d65f5236205dd23c6a8212627...	alway want sinc primari sc...	Fail
11	e03653dedd6f4e142f4aca1319...	love thing deal comput lik...	Fail
12	e7400496f1ce70cb62c2c44ca2...	seem interest tri	Pass

[42]: 63

Bag-of-Words

- Corpus
- Document
- Tokens

Corpus

[43]:
```
#
# A corpus is a collection of documents
# The total number of documents essential make up the corpus
len(var_ict1110_student_performance_text_example["MajorProgrammeMotivation"])
```

[43]: 63

Documents

[44]:
```
#
# Each individual docoument is accessible via its index and, or unique␣
↪identifier---StudentID
#
var_ict1110_student_performance_text_example.iloc[0][["StudentID",␣
↪"MajorProgrammeMotivation"]]
```

[44]:

	1
StudentID	742b8abe5776a6d942a92ce7dc...
MajorProgrammeMotivation	want acquir knowledg ict c...

[45]:
```
#
# Each individual docoument is accessible via its index and, or unique␣
↪identifier---StudentID
#

#
# Get sample StudentID
var_ict1110_student_performance_text_example.iloc[0]["StudentID"]

#
# Extract document associated with specified StudentID
var_ict1110_student_performance_text_example[var_ict1110_student_performance_text_example["Student
↪values

#
# Extract document as a string
var_ict1110_major_motivation_document0 =␣
↪var_ict1110_student_performance_text_example[var_ict1110_student_performance_text_example["Stud
↪values[0]

#
# Print out document
print(var_ict1110_major_motivation_document0)
```

[45]:
```
'742b8abe5776a6d942a92ce7dc7d84a0'
```

[45]:
```
array(['want acquir knowledg ict contribut technolog'], dtype=object)
```

```
want acquir knowledg ict contribut technolog
```

Tokens

[46]:
```
#
# Each document is made up of tokens: individual words making up the document
#
var_ict1110_major_motivation_document0.split()
```

[46]:
```
['want', 'acquir', 'knowledg', 'ict', 'contribut', 'technolog']
```

[47]:

```
#
# Essentially, our corpus is made up of documents, which are segmented into tokens
#
var_ict1110_student_performance_text_example["MajorProgrammeMotivation"].apply(str).
 ↪str.split()
```

[47]:

	MajorProgrammeMotivation
1	[want, acquir, knowledg, i...
2	[alway, want, ict, relat, ...
3	[result, met, requir]
4	[written, program, twice, ...
5	[part, parcel, ever, chang...
6	[alway, want, studi, program]
7	[use, comput, lot]
8	[know]
9	[know, behind, digit, world]
10	[alway, want, sinc, primar...
11	[love, thing, deal, comput...
12	[seem, interest, tri]
13	[aim, becom, program, ict,...
14	[want, part, technolog, wo...
15	[world, go, technolog, hen...
16	[ict, evolv, henc, give, t...
17	[closest, cours, comput, s...
18	[closest, cours, comput, s...
19	[market, want, knowledg, i...
20	[market, want, knowledg, i...
21	[want, knowledg, technolog...
22	[know, ict, live, world, e...
23	[knowledg, technolog, worl...
24	[fascin, comput, technolog]
25	[teach, other, improv, zam...
26	[live, technolog, world, d...
27	[inord, wider, understand,...
28	[order, lead, nation, worl...
29	[decid, pursu, b, ict, ed,...
30	[world, chang, term, techn...
31	[short, network, hardwar, ...
32	[comput, interest]
33	[comput, interest]
34	[though, choos, ict, first...
35	[learner, technolog]
93	[MISSING, DATA]
94	[MISSING, DATA]
95	[MISSING, DATA]
96	[MISSING, DATA]
97	[MISSING, DATA]
98	[MISSING, DATA]
99	[MISSING, DATA]

Continued on next page

	MajorProgrammeMotivation
100	[MISSING, DATA]
101	[MISSING, DATA]
102	[MISSING, DATA]
103	[MISSING, DATA]
104	[MISSING, DATA]
105	[MISSING, DATA]
106	[MISSING, DATA]
107	[MISSING, DATA]
108	[MISSING, DATA]
109	[MISSING, DATA]
110	[MISSING, DATA]
111	[MISSING, DATA]
112	[MISSING, DATA]
113	[MISSING, DATA]
114	[MISSING, DATA]
115	[MISSING, DATA]
116	[MISSING, DATA]
117	[MISSING, DATA]
118	[MISSING, DATA]
119	[MISSING, DATA]
120	[MISSING, DATA]

[48]:
```
#
# You can also split up documents into tokens using native Python
#
var_ict1110_student_major_motivation_corpus =␣
↪list(var_ict1110_student_performance_text_example["MajorProgrammeMotivation"])
```

[49]:
```
#
# Instead of this, using list comprehension
type(var_ict1110_student_major_motivation_corpus)
var_final_list = []
for var_doc in var_ict1110_student_major_motivation_corpus:
    var_final_list.append(var_doc.split())
#
var_final_list[:3]
```

[49]:
```
list
```

[49]:
```
[['want', 'acquir', 'knowledg', 'ict', 'contribut', 'technolog'],
 ['alway', 'want', 'ict', 'relat', 'program'],
 ['result', 'met', 'requir']]
```

[50]:
```
#
# For each document, loop to tokenize the document
var_ict1110_student_major_motivation_documents = [var_document.split() for␣
↪var_document in var_ict1110_student_major_motivation_corpus]
```

[51]:
```
var_ict1110_student_major_motivation_documents[:3]
```

[51]:
```
[['want', 'acquir', 'knowledg', 'ict', 'contribut', 'technolog'],
 ['alway', 'want', 'ict', 'relat', 'program'],
 ['result', 'met', 'requir']]
```

[52]:
```
#
# Corpus is now made up of two dimention vector: document and corresponding tokens



#
# Inspect corpus
print ("************************")
print (var_ict1110_student_major_motivation_documents[:30])



#
# Access desired document
print ("************************")
print ("First document in corpus: ", var_ict1110_student_major_motivation_documents[0])

#
# Access last token in the first document
print ("************************")
print ("Last token in first document: ",␣
↪var_ict1110_student_major_motivation_documents[0][-1])
```

```
************************
[['want', 'acquir', 'knowledg', 'ict', 'contribut', 'technolog'], ['alway', 'want',
'ict', 'relat', 'program'], ['result', 'met', 'requir'], ['written', 'program', 'twice',
'applic', 'form', 'man', 'collect', 'suggest', 'b', 'ict', 'ed'], ['part', 'parcel',
'ever', 'chang', 'develop', 'digit', 'world', 'take', 'part', 'zambia', 'vision',
'digit', 'zambia', 'sector', 'develop'], ['alway', 'want', 'studi', 'program'], ['use',
'comput', 'lot'], ['know'], ['know', 'behind', 'digit', 'world'], ['alway', 'want',
'sinc', 'primari', 'school'], ['love', 'thing', 'deal', 'comput', 'like', 'teach',
'peopl', 'time'], ['seem', 'interest', 'tri'], ['aim', 'becom', 'program', 'ict',
'sppecialist', 'day'], ['want', 'part', 'technolog', 'world', 'larg'], ['world', 'go',
'technolog', 'henc', 'abl', 'work', 'anywher', 'graduat'], ['ict', 'evolv', 'henc',
'give', 'time', 'learn', 'new', 'thing', 'relat', 'ict', 'help', 'think', 'big'],
['closest', 'cours', 'comput', 'scienc', 'outsid', 'school', 'natur', 'scienc'],
['closest', 'cours', 'comput', 'scienc', 'outsid', 'school', 'natur', 'scienc'],
['market', 'want', 'knowledg', 'ict', 'sinc', 'digit', 'world', 'would', 'also', 'love',
'studi', 'comput', 'engin'], ['market', 'want', 'knowledg', 'ict', 'sinc', 'digit',
'world', 'would', 'also', 'love', 'studi', 'comput', 'engin'], ['want', 'knowledg',
'technolog', 'comput'], ['know', 'ict', 'live', 'world', 'everyth', 'involv',
'technolog'], ['knowledg', 'technolog', 'world', 'live', 'today'], ['fascin', 'comput',
'technolog'], ['teach', 'other', 'improv', 'zambia', 'develop', 'nation', 'import',
'dream', 'becom', 'softwar', 'programm'], ['live', 'technolog', 'world', 'develop',
'everi', 'day', 'eager', 'learn'], ['inord', 'wider', 'understand', 'oper', 'comput',
'even', 'decid', 'studi', 'someth', 'els', 'ict', 'still', 'skill', 'ict', 'applic',
'cours', 'studi'], ['order', 'lead', 'nation', 'world', 'technolog'], ['decid', 'pursu',
'b', 'ict', 'ed', 'love', 'work', 'comput', 'peripher', 'devic', 'also', 'want', 'learn',
'softwar', 'develop', 'comput', 'hardwar', 'comput', 'network', 'someday', 'help',
'improv', 'comput', 'industri', 'zambia', 'develop', 'softwar', 'tailor', 'zambia',
```

```
'need', 'well', 'develop', 'comput', 'hardwar', 'one', 'thing', 'mind', 'start',
'comput', 'assembl', 'plant', 'right', 'zambia', 'later', 'also', 'manufactur',
'zambian', 'brand', 'comput'], ['world', 'chang', 'term', 'technolog', 'countri',
'zambia', 'still', 'remain', 'behind', 'adopt', 'technolog', 'therefor', 'want', 'one',
'take', 'zambia', 'anoth', 'level', 'term', 'technolog', 'pursu', 'degre']]
************************
First document in corpus:  ['want', 'acquir', 'knowledg', 'ict', 'contribut',
'technolog']
************************
Last token in first document:  technolog
```

[53]:
```
var_ict1110_student_major_motivation_documents[:4]
```

[53]:
```
[['want', 'acquir', 'knowledg', 'ict', 'contribut', 'technolog'],
 ['alway', 'want', 'ict', 'relat', 'program'],
 ['result', 'met', 'requir'],
 ['written',
  'program',
  'twice',
  'applic',
  'form',
  'man',
  'collect',
  'suggest',
  'b',
  'ict',
  'ed']]
```

Document Term Frequency

Corpus Document Terms

[54]:
```
#
# To manually build the document term frequencies for the corpus, we need to first␣
↪keep track of all unique terms
#

#
# Unrol all the documents and merge them into a single list
var_major_motivation_terms = []
for var_documents in var_ict1110_student_major_motivation_documents:
    var_major_motivation_terms += var_documents

print ("************************")
print ("Total terms from all documents: ", len(var_major_motivation_terms))
print (var_major_motivation_terms[:20]) # Spit out only a few entries

#
# To get unique terms, casting the list to a set will filter out duplicates---remember␣
↪that sets in Python contain unique entries
# One duplicates are removed, you can convert the set back to a list
```

```
var_major_motivation_terms = set(var_major_motivation_terms) # List to Set
var_major_motivation_terms = list(var_major_motivation_terms) # Set to List

print ("************************")
print ("Total unique terms from all documents: ", len(var_major_motivation_terms))
print (var_major_motivation_terms[:20])

#
# Sort entries for goog measure
var_major_motivation_terms.sort()

print ("************************")
print ("Total sorted unique terms from all documents: ",␣
↪len(var_major_motivation_terms))
print (var_major_motivation_terms[:20])
```

```
************************
Total terms from all documents:  359
['want', 'acquir', 'knowledg', 'ict', 'contribut', 'technolog', 'alway', 'want', 'ict',
'relat', 'program', 'result', 'met', 'requir', 'written', 'program', 'twice', 'applic',
'form', 'man']
************************
Total unique terms from all documents:  148
['decis', 'vision', 'countri', 'deal', 'start', 'decid', 'inord', 'learn', 'mainten',
'acquir', 'know', 'fascin', 'would', 'lot', 'requir', 'market', 'closest', 'primari',
'readili', 'contribut']
************************
Total sorted unique terms from all documents:  148
['DATA', 'MISSING', 'abl', 'acquir', 'adopt', 'aim', 'also', 'alway', 'anoth', 'anywher',
'applic', 'assembl', 'avail', 'b', 'becom', 'behind', 'big', 'brand', 'chang', 'choos']
```

Corpus Term Frequencies

[55]:
```
#
#
# Build frequency table for all terms in corpus to gain a sense of how many terms are␣
↪in the corpus

var_major_motivation_corpus_terms = {} # dictionary to hold term frequencies
for var_term in var_major_motivation_terms:
    var_term_frequency = 0
    for var_document in var_ict1110_student_major_motivation_documents:
        for var_doc_term in var_document:
            if var_doc_term == var_term:
                var_term_frequency += 1
    var_major_motivation_corpus_terms[var_term] = var_term_frequency

#
# Inspect term frequencies
print ("************************")
print (var_major_motivation_corpus_terms)
print ("************************")
```

```
var_major_motivation_corpus_terms.items()
```

```
************************
{'DATA': 28, 'MISSING': 28, 'abl': 1, 'acquir': 1, 'adopt': 1, 'aim': 1, 'also': 4,
'alway': 3, 'anoth': 1, 'anywher': 1, 'applic': 2, 'assembl': 1, 'avail': 1, 'b': 2,
'becom': 2, 'behind': 2, 'big': 1, 'brand': 1, 'chang': 2, 'choos': 1, 'cict': 1,
'closest': 2, 'collect': 1, 'comput': 18, 'contribut': 1, 'countri': 1, 'cours': 3,
'day': 2, 'deal': 1, 'decid': 2, 'decis': 1, 'degre': 1, 'develop': 8, 'devic': 1,
'digit': 5, 'dream': 1, 'eager': 1, 'ed': 2, 'els': 1, 'engin': 2, 'even': 1, 'ever': 1,
'everi': 1, 'everyth': 1, 'evolv': 1, 'fascin': 1, 'first': 1, 'form': 1, 'give': 1,
'go': 2, 'got': 1, 'graduat': 1, 'hardwar': 3, 'help': 2, 'henc': 2, 'ict': 13, 'import':
1, 'improv': 2, 'industri': 1, 'inord': 1, 'interest': 4, 'involv': 1, 'job': 1, 'know':
3, 'knowledg': 5, 'larg': 1, 'later': 1, 'lead': 1, 'learn': 4, 'learner': 1, 'level': 1,
'like': 1, 'live': 3, 'lot': 1, 'love': 4, 'mainten': 1, 'man': 1, 'manufactur': 1,
'market': 2, 'met': 1, 'mind': 1, 'nation': 2, 'natur': 2, 'need': 1, 'network': 2,
'new': 2, 'one': 2, 'oper': 1, 'order': 1, 'other': 1, 'outsid': 2, 'parcel': 1, 'part':
3, 'peopl': 1, 'peripher': 1, 'place': 1, 'plant': 1, 'primari': 1, 'program': 5,
'programm': 1, 'pursu': 2, 'readili': 1, 'relat': 2, 'remain': 1, 'requir': 1, 'result':
1, 'right': 1, 'school': 3, 'scienc': 4, 'sector': 1, 'seem': 1, 'short': 1, 'sinc': 4,
'skill': 1, 'softwar': 4, 'someday': 1, 'someth': 1, 'sppecialist': 1, 'start': 1,
'still': 2, 'studi': 5, 'suggest': 1, 'tailor': 1, 'take': 2, 'teach': 2, 'technolog':
13, 'term': 2, 'therefor': 1, 'thing': 3, 'think': 1, 'though': 1, 'thought': 1, 'time':
2, 'today': 1, 'tri': 1, 'twice': 1, 'understand': 1, 'use': 1, 'vision': 1, 'want': 10,
'well': 1, 'wider': 1, 'work': 2, 'world': 11, 'would': 2, 'written': 1, 'zambia': 8,
'zambian': 1}
************************
```

[55]:
```
dict_items([('DATA', 28), ('MISSING', 28), ('abl', 1), ('acquir', 1), ('adopt', 1),
('aim', 1), ('also', 4), ('alway', 3), ('anoth', 1), ('anywher', 1), ('applic', 2),
('assembl', 1), ('avail', 1), ('b', 2), ('becom', 2), ('behind', 2), ('big', 1),
('brand', 1), ('chang', 2), ('choos', 1), ('cict', 1), ('closest', 2), ('collect', 1),
('comput', 18), ('contribut', 1), ('countri', 1), ('cours', 3), ('day', 2), ('deal', 1),
('decid', 2), ('decis', 1), ('degre', 1), ('develop', 8), ('devic', 1), ('digit', 5),
('dream', 1), ('eager', 1), ('ed', 2), ('els', 1), ('engin', 2), ('even', 1), ('ever',
1), ('everi', 1), ('everyth', 1), ('evolv', 1), ('fascin', 1), ('first', 1), ('form',␣
↪1),
('give', 1), ('go', 2), ('got', 1), ('graduat', 1), ('hardwar', 3), ('help', 2),␣
↪('henc',
2), ('ict', 13), ('import', 1), ('improv', 2), ('industri', 1), ('inord', 1),
('interest', 4), ('involv', 1), ('job', 1), ('know', 3), ('knowledg', 5), ('larg', 1),
('later', 1), ('lead', 1), ('learn', 4), ('learner', 1), ('level', 1), ('like', 1),
('live', 3), ('lot', 1), ('love', 4), ('mainten', 1), ('man', 1), ('manufactur', 1),
('market', 2), ('met', 1), ('mind', 1), ('nation', 2), ('natur', 2), ('need', 1),
('network', 2), ('new', 2), ('one', 2), ('oper', 1), ('order', 1), ('other', 1),
('outsid', 2), ('parcel', 1), ('part', 3), ('peopl', 1), ('peripher', 1), ('place', 1),
('plant', 1), ('primari', 1), ('program', 5), ('programm', 1), ('pursu', 2), ('readili',
1), ('relat', 2), ('remain', 1), ('requir', 1), ('result', 1), ('right', 1), ('school',
3), ('scienc', 4), ('sector', 1), ('seem', 1), ('short', 1), ('sinc', 4), ('skill', 1),
('softwar', 4), ('someday', 1), ('someth', 1), ('sppecialist', 1), ('start', 1),
('still', 2), ('studi', 5), ('suggest', 1), ('tailor', 1), ('take', 2), ('teach', 2),
('technolog', 13), ('term', 2), ('therefor', 1), ('thing', 3), ('think', 1), ('though',
```

```
1), ('thought', 1), ('time', 2), ('today', 1), ('tri', 1), ('twice', 1), ('understand',
1), ('use', 1), ('vision', 1), ('want', 10), ('well', 1), ('wider', 1), ('work', 2),
('world', 11), ('would', 2), ('written', 1), ('zambia', 8), ('zambian', 1)])
```

Document Term Frequencies

For each of the corpus terms, count the the number of times the term exists in each document

[56]:
```
#
# Build a 2D list that will keep track of document frequencies
#
var_major_motivation_document_frequencies = [] # List to hold document frequencies
for var_document in var_ict1110_student_major_motivation_documents:
    var_term_document_frequencies = [] # List to keep track of term frequencies for␣
 ↪document
    for var_corpus_term in var_major_motivation_terms:
        var_term_frequency = 0 # Variable to keep track of occurance of term
        # Loop through terms in the document
        for var_document_term in var_document:
            if var_document_term == var_corpus_term:
                var_term_frequency += 1
        var_term_document_frequencies.append(var_term_frequency)
    var_major_motivation_document_frequencies.append(var_term_document_frequencies)


#
#
print ("************************")
print("Corpus terms")
print (var_major_motivation_terms)
#
#
print ("************************")
print ("First couple of document frequencies")
print (var_major_motivation_document_frequencies[:2])
```

```
************************
Corpus terms
['DATA', 'MISSING', 'abl', 'acquir', 'adopt', 'aim', 'also', 'alway', 'anoth', 'anywher',
'applic', 'assembl', 'avail', 'b', 'becom', 'behind', 'big', 'brand', 'chang', 'choos',
'cict', 'closest', 'collect', 'comput', 'contribut', 'countri', 'cours', 'day', 'deal',
'decid', 'decis', 'degre', 'develop', 'devic', 'digit', 'dream', 'eager', 'ed', 'els',
'engin', 'even', 'ever', 'everi', 'everyth', 'evolv', 'fascin', 'first', 'form', 'give',
'go', 'got', 'graduat', 'hardwar', 'help', 'henc', 'ict', 'import', 'improv', 'industri',
'inord', 'interest', 'involv', 'job', 'know', 'knowledg', 'larg', 'later', 'lead',
'learn', 'learner', 'level', 'like', 'live', 'lot', 'love', 'mainten', 'man',
'manufactur', 'market', 'met', 'mind', 'nation', 'natur', 'need', 'network', 'new',
'one', 'oper', 'order', 'other', 'outsid', 'parcel', 'part', 'peopl', 'peripher',
'place', 'plant', 'primari', 'program', 'programm', 'pursu', 'readili', 'relat',
'remain', 'requir', 'result', 'right', 'school', 'scienc', 'sector', 'seem', 'short',
'sinc', 'skill', 'softwar', 'someday', 'someth', 'sppecialist', 'start', 'still',
```

```
'studi', 'suggest', 'tailor', 'take', 'teach', 'technolog', 'term', 'therefor', 'thing',
'think', 'though', 'thought', 'time', 'today', 'tri', 'twice', 'understand', 'use',
'vision', 'want', 'well', 'wider', 'work', 'world', 'would', 'written', 'zambia',
'zambian']
************************
First couple of document frequencies
[[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]]
```

Document Term Frequencies: CountVectorizer

We can do what was previously manually done using scikit-learn's CountVectorizer, where every row will represent a different document and every column will represent a different word.

[57]:
```
#
# Create DataFrame for experimentation
#
var_ict1110_student_performance_text_example.head(3)
var_ict1110_student_major_motivation_corpus2 =␣
↪var_ict1110_student_performance_text_example[["StudentID",␣
↪"MajorProgrammeMotivation"]]
```

[57]:

	StudentID	MajorProgrammeMotivation	ExaminationStatus
1	742b8abe5776a6d942a92ce7dc...	want acquir knowledg ict c...	Pass
2	921855f753932de762b780405a...	alway want ict relat program	Pass
3	07f3ca235faaa1c9ad16facef5...	result met requir	Pass

[58]:
```
var_ict1110_student_major_motivation_corpus2.head(1)
```

[58]:

	StudentID	MajorProgrammeMotivation
1	742b8abe5776a6d942a92ce7dc...	want acquir knowledg ict c...

[59]:
```
#
# Use CountVectorizer object to derive corpus document frequencies
# from sklearn.feature_extraction.text import CountVectorizer
#
# NOTE: Additional parameters can be fed to CountVectorizer to improve accuracy of␣
↪models. More on that later
# var_major_motivation_vectoriser_cv = CountVectorizer(stop_words='english', max_df =␣
↪0.90, min_df = 0.01, ngram_range=(1, 3))
#
```

```
#
var_major_motivation_vectoriser_cv = CountVectorizer()
```

[60]:
```
#
# Fit and Transform applicable DataFrame column
#
var_major_motivation_vectoriser_cv.
 ↪fit_transform(var_ict1110_student_major_motivation_corpus2["MajorProgrammeMotivation"])

#
# Create variable to hold transform text
#
var_major_motivation_vectoriser_cv_data = var_major_motivation_vectoriser_cv.
 ↪fit_transform(var_ict1110_student_major_motivation_corpus2["MajorProgrammeMotivation"])
```

[60]:
```
<63x147 sparse matrix of type '<class 'numpy.longlong'>'
        with 331 stored elements in Compressed Sparse Row format>
```

[61]:
```
var_major_motivation_vectoriser_cv.vocabulary_
var_major_motivation_vectoriser_cv_data[0].toarray()
```

[61]:
```
{'want': 138,
 'acquir': 1,
 'knowledg': 62,
 'ict': 53,
 'contribut': 21,
 'technolog': 124,
 'alway': 5,
 'relat': 101,
 'program': 97,
 'result': 104,
 'met': 77,
 'requir': 103,
 'written': 144,
 'twice': 134,
 'applic': 8,
 'form': 45,
 'man': 74,
 'collect': 19,
 'suggest': 120,
 'ed': 35,
 'part': 91,
 'parcel': 90,
 'ever': 39,
 'chang': 15,
 'develop': 30,
 'digit': 32,
 'world': 142,
 'take': 122,
 'zambia': 145,
 'vision': 137,
```

```
'sector': 108,
'studi': 119,
'use': 136,
'comput': 20,
'lot': 71,
'know': 61,
'behind': 12,
'sinc': 111,
'primari': 96,
'school': 106,
'love': 72,
'thing': 127,
'deal': 26,
'like': 69,
'teach': 123,
'peopl': 92,
'time': 131,
'seem': 109,
'interest': 58,
'tri': 133,
'aim': 3,
'becom': 11,
'sppecialist': 116,
'day': 25,
'larg': 63,
'go': 47,
'henc': 52,
'abl': 0,
'work': 141,
'anywher': 7,
'graduat': 49,
'evolv': 42,
'give': 46,
'learn': 66,
'new': 84,
'help': 51,
'think': 128,
'big': 13,
'closest': 18,
'cours': 23,
'scienc': 107,
'outsid': 89,
'natur': 81,
'market': 76,
'would': 143,
'also': 4,
'engin': 37,
'live': 70,
'everyth': 41,
'involv': 59,
'today': 132,
```

```
'fascin': 43,
'other': 88,
'improv': 55,
'nation': 80,
'import': 54,
'dream': 33,
'softwar': 113,
'programm': 98,
'everi': 40,
'eager': 34,
'inord': 57,
'wider': 140,
'understand': 135,
'oper': 86,
'even': 38,
'decid': 27,
'someth': 115,
'els': 36,
'still': 118,
'skill': 112,
'order': 87,
'lead': 65,
'pursu': 99,
'peripher': 93,
'devic': 31,
'hardwar': 50,
'network': 83,
'someday': 114,
'industri': 56,
'tailor': 121,
'need': 82,
'well': 139,
'one': 85,
'mind': 78,
'start': 117,
'assembl': 9,
'plant': 95,
'right': 105,
'later': 64,
'manufactur': 75,
'zambian': 146,
'brand': 14,
'term': 125,
'countri': 22,
'remain': 102,
'adopt': 2,
'therefor': 126,
'anoth': 6,
'level': 68,
'degre': 29,
'short': 110,
```

```
 'mainten': 73,
 'cict': 17,
 'though': 129,
 'choos': 16,
 'first': 44,
 'place': 94,
 'thought': 130,
 'job': 60,
 'readili': 100,
 'avail': 10,
 'got': 48,
 'decis': 28,
 'learner': 67,
 'missing': 79,
 'data': 24}
```

[61]:
```
array([[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]])
```

[62]:
```
#
# CountVectorizer feature names
#
var_major_motivation_vectoriser_cv.get_feature_names()
```

[62]:
```
['abl',
 'acquir',
 'adopt',
 'aim',
 'also',
 'alway',
 'anoth',
 'anywher',
 'applic',
 'assembl',
 'avail',
 'becom',
 'behind',
 'big',
 'brand',
 'chang',
 'choos',
 'cict',
 'closest',
 'collect',
 'comput',
 'contribut',
```

```
'countri',
'cours',
'data',
'day',
'deal',
'decid',
'decis',
'degre',
'develop',
'devic',
'digit',
'dream',
'eager',
'ed',
'els',
'engin',
'even',
'ever',
'everi',
'everyth',
'evolv',
'fascin',
'first',
'form',
'give',
'go',
'got',
'graduat',
'hardwar',
'help',
'henc',
'ict',
'import',
'improv',
'industri',
'inord',
'interest',
'involv',
'job',
'know',
'knowledg',
'larg',
'later',
'lead',
'learn',
'learner',
'level',
'like',
'live',
'lot',
'love',
```

```
'mainten',
'man',
'manufactur',
'market',
'met',
'mind',
'missing',
'nation',
'natur',
'need',
'network',
'new',
'one',
'oper',
'order',
'other',
'outsid',
'parcel',
'part',
'peopl',
'peripher',
'place',
'plant',
'primari',
'program',
'programm',
'pursu',
'readili',
'relat',
'remain',
'requir',
'result',
'right',
'school',
'scienc',
'sector',
'seem',
'short',
'sinc',
'skill',
'softwar',
'someday',
'someth',
'sppecialist',
'start',
'still',
'studi',
'suggest',
'tailor',
'take',
'teach',
```

```
 'technolog',
 'term',
 'therefor',
 'thing',
 'think',
 'though',
 'thought',
 'time',
 'today',
 'tri',
 'twice',
 'understand',
 'use',
 'vision',
 'want',
 'well',
 'wider',
 'work',
 'world',
 'would',
 'written',
 'zambia',
 'zambian']
```

[63]:
```
#
# Inspect constructed data structure
type(var_major_motivation_vectoriser_cv_data[5])

#
# Convert sparse matrix into array to be used to merge with DataFrame
var_major_motivation_vectoriser_cv_data[5].toarray()
```

[63]:
```
scipy.sparse.csr.csr_matrix
```

[63]:
```
array([[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0]])
```

[64]:
```
#
# Construct a DataFrame with transformed entries
#
var_major_motivation_vectoriser_cv_data_tf = pd.
 ↪DataFrame(var_major_motivation_vectoriser_cv_data.toarray(),␣
 ↪columns=var_major_motivation_vectoriser_cv.get_feature_names())
```

[65]:
```
var_major_motivation_vectoriser_cv_data_tf.head(12)
```

[65]:

	abl	acquir	adopt	aim	also	alway	anoth	anywher	applic	assembl	avail	becom	behind	bi
0	0	1	0	0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	1	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	1	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	0	0	0	0	0	1	0	0	0	0	0	0	0	
6	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	0	0	0	0	0	0	0	0	0	1	
9	0	0	0	0	0	1	0	0	0	0	0	0	0	
10	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	0	0	0	0	0	0	0	0	0	0	0	0	0	

TF-IDF

Exercise: Implement TF-IDF

Document Term Frequencies: TfidfVectorizer

Similar to CountVectorzer, we can take advantage of scikit-learn's TfidfVectorizer, where every row will represent a different document and every column will represent a TF-IDF value for each term.

[66]:
```
#
# Create DataFrame for experimentation
#
var_ict1110_student_performance_text_example.head(3)
var_ict1110_student_major_motivation_corpus3 =␣
↪var_ict1110_student_performance_text_example[["StudentID",␣
↪"MajorProgrammeMotivation"]]
```

[66]:

	StudentID	MajorProgrammeMotivation	ExaminationStatus
1	742b8abe5776a6d942a92ce7dc...	want acquir knowledg ict c...	Pass
2	921855f753932de762b780405a...	alway want ict relat program	Pass
3	07f3ca235faaa1c9ad16facef5...	result met requir	Pass

[67]:
```
var_ict1110_student_major_motivation_corpus3.head(1)
```

[67]:

	StudentID	MajorProgrammeMotivation
1	742b8abe5776a6d942a92ce7dc...	want acquir knowledg ict c...

[68]:
```
#
# Use TfidfVectorizer object to derive corpus document TF-IDF values
# from sklearn.feature_extraction.text import TfidfVectorizer
#
# NOTE: Additional parameters can be fed to TfidfVectorizer to improve accuracy of␣
↪models. More on that later
```

```
# var_major_motivation_vectoriser_tfidfv = TfidfVectorizer(stop_words='english',␣
↪max_df = 0.90, min_df = 0.01, ngram_range=(1, 3))
#
#
var_major_motivation_vectoriser_tfidfv = TfidfVectorizer()
```

[69]:
```
#
# Fit and Transform applicable DataFrame column
#
var_major_motivation_vectoriser_tfidfv.
↪fit_transform(var_ict1110_student_major_motivation_corpus3["MajorProgrammeMotivation"])

#
# Create variable to hold transform text
#
var_major_motivation_vectoriser_tfidfv_data = var_major_motivation_vectoriser_tfidfv.
↪fit_transform(var_ict1110_student_major_motivation_corpus3["MajorProgrammeMotivation"])
```

[69]:
```
<63x147 sparse matrix of type '<class 'numpy.float64'>'
        with 331 stored elements in Compressed Sparse Row format>
```

[70]:
```
#
# CountVectorizer feature names
#
var_major_motivation_vectoriser_tfidfv.get_feature_names()
```

[70]:
```
['abl',
 'acquir',
 'adopt',
 'aim',
 'also',
 'alway',
 'anoth',
 'anywher',
 'applic',
 'assembl',
 'avail',
 'becom',
 'behind',
 'big',
 'brand',
 'chang',
 'choos',
 'cict',
 'closest',
 'collect',
 'comput',
 'contribut',
 'countri',
 'cours',
 'data',
```

```
'day',
'deal',
'decid',
'decis',
'degre',
'develop',
'devic',
'digit',
'dream',
'eager',
'ed',
'els',
'engin',
'even',
'ever',
'everi',
'everyth',
'evolv',
'fascin',
'first',
'form',
'give',
'go',
'got',
'graduat',
'hardwar',
'help',
'henc',
'ict',
'import',
'improv',
'industri',
'inord',
'interest',
'involv',
'job',
'know',
'knowledg',
'larg',
'later',
'lead',
'learn',
'learner',
'level',
'like',
'live',
'lot',
'love',
'mainten',
'man',
'manufactur',
```

```
'market',
'met',
'mind',
'missing',
'nation',
'natur',
'need',
'network',
'new',
'one',
'oper',
'order',
'other',
'outsid',
'parcel',
'part',
'peopl',
'peripher',
'place',
'plant',
'primari',
'program',
'programm',
'pursu',
'readili',
'relat',
'remain',
'requir',
'result',
'right',
'school',
'scienc',
'sector',
'seem',
'short',
'sinc',
'skill',
'softwar',
'someday',
'someth',
'sppecialist',
'start',
'still',
'studi',
'suggest',
'tailor',
'take',
'teach',
'technolog',
'term',
'therefor',
```

```
 'thing',
 'think',
 'though',
 'thought',
 'time',
 'today',
 'tri',
 'twice',
 'understand',
 'use',
 'vision',
 'want',
 'well',
 'wider',
 'work',
 'world',
 'would',
 'written',
 'zambia',
 'zambian']
```

[71]:
```
#
# Inspect constructed data structure
type(var_major_motivation_vectoriser_tfidfv_data[5])

#
# Convert sparse matrix into array to be used to merge with DataFrame
var_major_motivation_vectoriser_tfidfv_data[5].toarray()
```

[71]:
```
scipy.sparse.csr.csr_matrix
```

[71]:
```
array([[0.        , 0.        , 0.        , 0.        , 0.        ,
        0.55750239, 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.        , 0.        , 0.        ,
        0.        , 0.        , 0.49758391, 0.        , 0.        ,
```

```
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        , 0.        , 0.        , 0.52452687,
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        , 0.        , 0.4080109 , 0.        ,
0.        , 0.        , 0.        , 0.        , 0.        ,
0.        , 0.        ]])
```

[72]:
```
#
# Construct a DataFrame with transformed entries
#
var_major_motivation_vectoriser_tfidfv_data_tfidf = pd.
 ↪DataFrame(var_major_motivation_vectoriser_tfidfv_data.toarray(),␣
 ↪columns=var_major_motivation_vectoriser_tfidfv.get_feature_names())
```

[73]:
```
var_major_motivation_vectoriser_tfidfv_data_tfidf.head(12)
```

[73]:

	abl	acquir	adopt	aim	also	alway	anoth	anywher	applic	assembl	avail	becom	beh
0	0.0	0.522151	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
1	0.0	0.000000	0.0	0.0	0.0	0.500443	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
2	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
3	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.310391	0.0	0.0	0.0	0.000
4	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
5	0.0	0.000000	0.0	0.0	0.0	0.557502	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
6	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
7	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
8	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.571
9	0.0	0.000000	0.0	0.0	0.0	0.455391	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
10	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000
11	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.000000	0.0	0.0	0.0	0.000

Example Dataset: TF vs TF-IDF

[74]:
```
print ("***********************")
var_major_motivation_vectoriser_cv_data_tf.head(3)
print ("***********************")
var_major_motivation_vectoriser_tfidfv_data_tfidf.head(3)
```

****

[74]:

	abl	acquir	adopt	aim	also	alway	anoth	anywher	applic	assembl	avail	becom	behind	big
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0

****

[74]:

	abl	acquir	adopt	aim	also	alway	anoth	anywher	applic	assembl	avail	becom	behind
0	0.0	0.522151	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.000000	0.0	0.0	0.0	0.500443	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Putting Everything Together

The output for the data transformation phases are input features tranformed into a numeric form that the estimator will be able to work with. In the case of our examples, all selected attributes must be transformed * Initial Survey Dataset * StudentID (N/A—Field used as unique identifier) * HomeTown—Categorical to One-Hot-Encoding * MinorProgrammeMotivation—Text to TF/TF-IDF * MajorProgrammeMotivation—Text to TF/TF-IDF * DidComputerStudies—Categorical to One-Hot-Encoding * HasComputerTraining—Categorical to One-Hot-Encoding * ExperienceWithComputers—Ordinal to Numeric Encoding * HasComputerAccess—Categorical to On-Hot-Encoding * Student Demographics Dataset * StudentAge—Numeric to Unchanged * Gender—Categorical to One-Hot-Encoding * MinorDescription—Categorical to One-Hot-Encoding * Sponsor—Categorical to One-Hot-Encoding * Accommodated—Categorical to One-Hot-Encoding * Assessment Scores Dataset * XPCT—Numeric to Unchanged * XStatus—Categorical to Normal Encoding * XGrade—Categorical to Normal Encoding * XGPA—Categorical to Normal Encoding

[75]:
```
#
# Inspect original DataFrame
#
var_ict1110_student_performance_dataframe_selected.columns

#
# DataFrame to hold final input vector
#
var_ict1110_student_performance_dataframe_transformed =␣
↪var_ict1110_student_performance_dataframe_selected
```

[75]:
```
Index(['StudentID', 'StudentLocation', 'MinorProgrammeMotivation',
       'MajorProgrammeMotivation', 'DidComputerStudies', 'HasComputerTraining',
       'ExperienceWithComputers', 'HasComputerAccess', 'StudentAge', 'Gender',
       ...
       'Test3Status', 'Test3Grade', 'Test3GPA', 'Test4Status', 'Test4Grade',
       'Test4GPA', 'ExaminationPCT', 'ExaminationStatus', 'ExaminationGrade',
       'ExaminationGPA'],
      dtype='object', length=114)
```

Dataset #1: Initial Survey

StudentID (N/A—Field used as unique identifier)

[76]:
```
# Nothing to do here
```

HomeTown—Categorical to One-Hot-Encoding

[77]:
```
var_ict1110_student_performance_dataframe_transformed["StudentLocation"].unique()
var_ict1110_student_performance_dataframe_transformed["StudentLocation"].head(3)
```

[77]:
```
array(['Copperbelt', 'Western', 'Lusaka', 'Central', 'Muchinga',
       'MISSING DATA', 'Eastern', 'Luapula', 'Southern', 'NorthWestern'],
      dtype=object)
```

[77]:

	StudentLocation
1	Copperbelt
2	Western
3	Lusaka

[78]:
```
var_ict1110_student_performance_dataframe_transformed.columns

for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
    print (var_index, var_item)
```

[78]:
```
Index(['StudentID', 'StudentLocation', 'MinorProgrammeMotivation',
       'MajorProgrammeMotivation', 'DidComputerStudies', 'HasComputerTraining',
       'ExperienceWithComputers', 'HasComputerAccess', 'StudentAge', 'Gender',
       ...
       'Test3Status', 'Test3Grade', 'Test3GPA', 'Test4Status', 'Test4Grade',
       'Test4GPA', 'ExaminationPCT', 'ExaminationStatus', 'ExaminationGrade',
       'ExaminationGPA'],
      dtype='object', length=114)
```

```
0 StudentID
1 StudentLocation
2 MinorProgrammeMotivation
3 MajorProgrammeMotivation
4 DidComputerStudies
5 HasComputerTraining
6 ExperienceWithComputers
7 HasComputerAccess
8 StudentAge
9 Gender
10 MinorDescription
11 Sponsor
12 Accommodated
13 Status
14 Quiz1PCT
15 Quiz2PCT
16 Quiz3PCT
17 Quiz4PCT
18 Quiz5PCT
19 Quiz6PCT
20 Quiz7PCT
```

```
21 Quiz8PCT
22 Quiz9PCT
23 Quiz10PCT
24 Quiz11PCT
25 Quiz12PCT
26 Quiz13PCT
27 Quiz14PCT
28 Quiz15PCT
29 Quiz16PCT
30 Quiz17PCT
31 Quiz18PCT
32 Quiz19PCT
33 Quiz20PCT
34 Quiz1Status
35 Quiz1Grade
36 Quiz1GPA
37 Quiz2Status
38 Quiz2Grade
39 Quiz2GPA
40 Quiz3Status
41 Quiz3Grade
42 Quiz3GPA
43 Quiz4Status
44 Quiz4Grade
45 Quiz4GPA
46 Quiz5Status
47 Quiz5Grade
48 Quiz5GPA
49 Quiz6Status
50 Quiz6Grade
51 Quiz6GPA
52 Quiz7Status
53 Quiz7Grade
54 Quiz7GPA
55 Quiz8Status
56 Quiz8Grade
57 Quiz8GPA
58 Quiz9Status
59 Quiz9Grade
60 Quiz9GPA
61 Quiz10Status
62 Quiz10Grade
63 Quiz10GPA
64 Quiz11Status
65 Quiz11Grade
66 Quiz11GPA
67 Quiz12Status
68 Quiz12Grade
69 Quiz12GPA
70 Quiz13Status
71 Quiz13Grade
72 Quiz13GPA
```

```
73 Quiz14Status
74 Quiz14Grade
75 Quiz14GPA
76 Quiz15Status
77 Quiz15Grade
78 Quiz15GPA
79 Quiz16Status
80 Quiz16Grade
81 Quiz16GPA
82 Quiz17Status
83 Quiz17Grade
84 Quiz17GPA
85 Quiz18Status
86 Quiz18Grade
87 Quiz18GPA
88 Quiz19Status
89 Quiz19Grade
90 Quiz19GPA
91 Quiz20Status
92 Quiz20Grade
93 Quiz20GPA
94 Test1PCT
95 Test2PCT
96 Test3PCT
97 Test4PCT
98 Test1Status
99 Test1Grade
100 Test1GPA
101 Test2Status
102 Test2Grade
103 Test2GPA
104 Test3Status
105 Test3Grade
106 Test3GPA
107 Test4Status
108 Test4Grade
109 Test4GPA
110 ExaminationPCT
111 ExaminationStatus
112 ExaminationGrade
113 ExaminationGPA
```

[79]:
```
#
# Create final StudentLocation DataFrame
#
var_ict1110_student_performance_dataframe_transformed_studentLocation = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["StudentLocation"])
```

[80]:
```
#
#
print ("************************")
```

```
var_ict1110_student_performance_dataframe_transformed_studentLocation.columns

#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_studentLocation.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
************************
```

[80]:
```
Index(['Central', 'Copperbelt', 'Eastern', 'Luapula', 'Lusaka', 'MISSING DATA',
       'Muchinga', 'NorthWestern', 'Southern', 'Western'],
      dtype='object')
```

```
************************
```

[80]:

	Central	Copperbelt	Eastern	Luapula	Lusaka	MISSING DATA	Muchinga	NorthWestern	Southern
1	0	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	1	0	0	0	0

MinorProgrammeMotivation—Text to TF/TF-IDF

[81]:
```
#
# Create TfidfVectorizer object
var_minor_motivation_vectoriser_tfidfv = TfidfVectorizer()

#
# Fit and Transform applicable DataFrame column
var_minor_motivation_vectoriser_tfidfv.
↪fit_transform(var_ict1110_student_performance_dataframe_transformed["MinorProgrammeMotivation"]

#
# Create variable to hold transform text
var_minor_motivation_vectoriser_tfidfv_data = var_minor_motivation_vectoriser_tfidfv.
↪fit_transform(var_ict1110_student_performance_dataframe_transformed["MinorProgrammeMotivation"]

#
# Construct a DataFrame with transformed entries
var_minor_motivation_vectoriser_tfidfv_data_tfidf = pd.
↪DataFrame(var_minor_motivation_vectoriser_tfidfv_data.toarray(),␣
↪columns=var_minor_motivation_vectoriser_tfidfv.get_feature_names())

#
```

```
#
var_minor_motivation_vectoriser_tfidfv_data_tfidf.head(3)
```

[81]:
```
<63x128 sparse matrix of type '<class 'numpy.float64'>'
        with 252 stored elements in Compressed Sparse Row format>
```

[81]:

	academ	addit	affair	also	alway	anchor	appreci	apprecit	art	avail	avoid	belief	believ
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.652968	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0

[82]:
```
#
# Create final MinorDescription DataFrame
#
var_ict1110_student_performance_dataframe_transformed_minorDescription =␣
↪var_minor_motivation_vectoriser_tfidfv_data_tfidf
```

[83]:
```
var_ict1110_student_performance_dataframe_transformed_minorDescription.head(3)
```

[83]:

	academ	addit	affair	also	alway	anchor	appreci	apprecit	art	avail	avoid	belief	believ
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.652968	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.0

MajorProgrammeMotivation—Text to TF/TF-IDF

[84]:
```
#
# Create TfidfVectorizer object
var_major_motivation_vectoriser_tfidfv = TfidfVectorizer()

#
# Fit and Transform applicable DataFrame column
var_major_motivation_vectoriser_tfidfv.
↪fit_transform(var_ict1110_student_performance_dataframe_transformed["MajorProgrammeMotivation"]

#
# Create variable to hold transform text
var_major_motivation_vectoriser_tfidfv_data = var_major_motivation_vectoriser_tfidfv.
↪fit_transform(var_ict1110_student_performance_dataframe_transformed["MajorProgrammeMotivation"]

#
# Construct a DataFrame with transformed entries
var_major_motivation_vectoriser_tfidfv_data_tfidf = pd.
↪DataFrame(var_major_motivation_vectoriser_tfidfv_data.toarray(),␣
↪columns=var_major_motivation_vectoriser_tfidfv.get_feature_names())

#
#
```

```
var_major_motivation_vectoriser_tfidfv_data_tfidf.head(3)
```

[84]:
```
<63x147 sparse matrix of type '<class 'numpy.float64'>'
        with 331 stored elements in Compressed Sparse Row format>
```

[84]:

	abl	acquir	adopt	aim	also	alway	anoth	anywher	applic	assembl	avail	becom	behind
0	0.0	0.522151	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.000000	0.0	0.0	0.0	0.500443	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0

[85]:
```
#
# Create final MajorDescription DataFrame
#
var_ict1110_student_performance_dataframe_transformed_majorDescription =␣
 ↪var_major_motivation_vectoriser_tfidfv_data_tfidf
```

[86]:
```
var_ict1110_student_performance_dataframe_transformed_majorDescription.head(3)
```

[86]:

	abl	acquir	adopt	aim	also	alway	anoth	anywher	applic	assembl	avail	becom	behind
0	0.0	0.522151	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	0.000000	0.0	0.0	0.0	0.500443	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.000000	0.0	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0

DidComputerStudies—Categorical to One-Hot-Encoding

[87]:
```
#
# Create final StudentLocation DataFrame
#
var_ict1110_student_performance_dataframe_transformed_didComputerStudies = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["DidComputerStudies"])
```

[88]:
```
#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_didComputerStudies.columns

#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_didComputerStudies.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
************************
```

[88]:
```
Index(['MISSING DATA', 'No', 'Yes'], dtype='object')
```

```
************************
```

[88]:

	MISSING DATA	No	Yes
1	0	1	0
2	0	1	0
3	0	1	0

HasComputerTraining—Categorical to One-Hot-Encoding

[89]:
```
#
# Create final StudentLocation DataFrame
#
var_ict1110_student_performance_dataframe_transformed_hasComputerTraining = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["HasComputerTraining"])
```

[90]:
```
#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_hasComputerTraining.columns

#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_hasComputerTraining.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
************************
```

[90]:
```
Index(['MISSING DATA', 'No', 'Yes'], dtype='object')
```

```
************************
```

[90]:

	MISSING DATA	No	Yes
1	0	1	0
2	0	1	0
3	0	1	0

ExperienceWithComputers—Ordinal to Numeric Encoding

[91]:
```
var_ict1110_student_performance_dataframe_transformed_experienceWithComputers =␣
↪var_ict1110_student_performance_dataframe_transformed[["ExperienceWithComputers"]]
```

[92]:
```
#
# Create dictionary mapping of ordinal values
var_computing_experience_dictionary = {
    'No Experience': 1,
    'Less than 1 year': 2,
    '1 to 2 years': 3,
    'More than 5 years': 4,
    'MISSING': 5
}
```

[93]:
```
#
#
var_ict1110_student_performance_dataframe_transformed_experienceWithComputers.head(3)

#
# Create new DataFrame column with encoded values
var_ict1110_student_performance_dataframe_transformed_experienceWithComputers["ExperienceWithCompu
↪= var_ict1110_student_performance_ordinal_example1["ExperienceWithComputers"].
↪map(var_computing_experience_dictionary)

#
#
var_ict1110_student_performance_dataframe_transformed_experienceWithComputers.head(3)
```

[93]:

	ExperienceWithComputers
1	1 to 2 years
2	No Experience
3	Less than 1 year

```
/home/lightonphiri/.local/lib/python3.6/site-packages/ipykernel_launcher.py:7:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  import sys
```

[93]:

	ExperienceWithComputers
1	3.0
2	1.0
3	2.0

HasComputerAccess—Categorical to On-Hot-Encoding

[94]:
```
#
# Create final HasComputerAccess DataFrame
#
var_ict1110_student_performance_dataframe_transformed_hasComputerAccess  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["HasComputerAccess"])
```

[95]:
```
#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_hasComputerAccess.columns

#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_hasComputerAccess.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
************************
```

[95]:
```
Index(['MISSING DATA', 'No', 'Yes'], dtype='object')
```

```
************************
```

[95]:

	MISSING DATA	No	Yes
1	0	0	1
2	0	0	1
3	0	0	1

Dataset #2: Student Demographics

StudentAge—Numeric to Unchanged

[96]:
```
var_ict1110_student_performance_dataframe_transformed["StudentAge"].unique()
```

[96]:
```
array([20., 19., 22., 24., 18., 21., 17., 36., 29., 25., 23., 26.])
```

[97]:
```
## Nothing to do here: attribute always numeric
# Perhaps this could be represented using bands?
```

Gender—Categorical to One-Hot-Encoding

[98]:
```
#
# Unique Gender entries
var_ict1110_student_performance_dataframe_transformed["Gender"].unique()
```

[98]:
```
array(['F', 'M'], dtype=object)
```

[99]:
```
#
# Create final HasComputerAccess DataFrame
#
var_ict1110_student_performance_dataframe_transformed_gender  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["Gender"])
```

[100]:
```
#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_gender.columns

#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_gender.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
************************
```

[100]:
```
Index(['F', 'M'], dtype='object')
```

```
************************
```

[100]:

	F	M
1	1	0
2	0	1
3	0	1

MinorDescription—Categorical to One-Hot-Encoding

[101]:
```
#
# Unique Gender entries
var_ict1110_student_performance_dataframe_transformed["MinorDescription"].unique()
```

[101]:
```
array(['MATHEMATICS', 'FRENCH', 'RELIGIOUS STUDIES', 'CIVIC EDUCATION',
       'MISSING DATA', 'ENGLISH', 'HISTORY', 'ART AND DESIGN STUDIES',
       'GEOGRAPHY'], dtype=object)
```

[102]:
```
#
# Create final MinorDescription DataFrame
#
var_ict1110_student_performance_dataframe_transformed_minorDescription  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["MinorDescription"])
```

[103]:
```
#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_minorDescription.columns

#
#
print ("************************")
var_ict1110_student_performance_dataframe_transformed_minorDescription.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
************************
```

[103]:
```
Index(['ART AND DESIGN STUDIES', 'CIVIC EDUCATION', 'ENGLISH', 'FRENCH',
       'GEOGRAPHY', 'HISTORY', 'MATHEMATICS', 'MISSING DATA',
       'RELIGIOUS STUDIES'],
      dtype='object')
```

```
************************
```

[103]:

	ART AND DESIGN STUDIES	CIVIC EDUCATION	ENGLISH	FRENCH	GEOGRAPHY	HISTORY	M
1	0	0	0	0	0	0	
2	0	0	0	1	0	0	
3	0	0	0	0	0	0	

Sponsor—Categorical to One-Hot-Encoding

[104]:
```
#
# Unique Sponsor entries
var_ict1110_student_performance_dataframe_transformed["Sponsor"].unique()
```

[104]:
```
array(['GRZ-FULLY SPONSORED', 'GRZ - 75 PERCENT', 'SELF-SPONSORED',
       'TUITION WAIVER(DEPENDANTS)'], dtype=object)
```

[105]:
```
#
# Create final Sponsor DataFrame
#
```

```
var_ict1110_student_performance_dataframe_transformed_sponsor  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["Sponsor"])
```

[106]:
```
#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_sponsor.columns

#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_sponsor.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
 ↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

```
***********************
```

[106]:
```
Index(['GRZ - 75 PERCENT', 'GRZ-FULLY SPONSORED', 'SELF-SPONSORED',
       'TUITION WAIVER(DEPENDANTS)'],
      dtype='object')
```

```
***********************
```

[106]:

	GRZ - 75 PERCENT	GRZ-FULLY SPONSORED	SELF-SPONSORED	TUITION WAIVER(DEPENDANTS)
1	0	1	0	0
2	0	1	0	0
3	0	1	0	0

Accommodated—Categorical to One-Hot-Encoding

[107]:
```
#
# Unique Accommodated entries
var_ict1110_student_performance_dataframe_transformed["Accommodated"].unique()
```

[107]:
```
array(['No', 'Yes'], dtype=object)
```

[108]:
```
#
# Create final HasComputerAccess DataFrame
#
var_ict1110_student_performance_dataframe_transformed_accommodated  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["Accommodated"])
```

[109]:
```
#
#
print ("***********************")
```

```
var_ict1110_student_performance_dataframe_transformed_accommodated.columns

#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_accommodated.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

****

[109]: Index(['No', 'Yes'], dtype='object')

****

[109]:

	No	Yes
1	1	0
2	0	1
3	1	0

[110]:
```
#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_accommodated.columns

#
#
print ("***********************")
var_ict1110_student_performance_dataframe_transformed_accommodated.head(3)

#
#
#####var_ict1110_student_performance_dataframe_transformed.head(2).T
#####for var_index, var_item in␣
↪enumerate(var_ict1110_student_performance_dataframe_transformed.columns):
#####    print (var_index, var_item)
```

****

[110]: Index(['No', 'Yes'], dtype='object')

****

[110]:

	No	Yes
1	1	0
2	0	1
3	1	0

Dataset #3: Assessment Scores

In essense, the assessment score would be the dependent variables, although some assessments, such as quizzes can be used as independent variables; for instance: * Using average quiz scores to determine if a student passes of fails a course * Using specific topics covered in the quizzes to determine if the student passes or

Note: For the purposes of using the assessment scores as examples, we will just work with examination scores.

XPCT—Numeric to Unchanged

[111]:
```
var_ict1110_student_performance_dataframe_transformed[["ExaminationPCT"]].head(3)
```

[111]:

	ExaminationPCT
1	56.0
2	85.0
3	64.0

[112]:
```
## Nothing to do here: attribute already in numeric form
```

XStatus—Categorical to Normal Encoding

[113]:
```
var_ict1110_student_performance_dataframe_transformed[["ExaminationStatus"]].head(3)
```

[113]:

	ExaminationStatus
1	Pass
2	Pass
3	Pass

[114]:
```
#
# Create final XStatus DataFrame
#
var_ict1110_student_performance_dataframe_transformed_examinationStatus  = pd.
↪get_dummies(var_ict1110_student_performance_dataframe_transformed["ExaminationStatus"])
```

[115]:
```
#
# Inspect encoded column
var_ict1110_student_performance_dataframe_transformed_examinationStatus.head(3)
```

[115]:

	Fail	Pass
1	0	1
2	0	1
3	0	1

XGrade—Categorical to Normal Encoding

[116]:
```
var_ict1110_student_performance_dataframe_transformed[["ExaminationGrade"]].head(3)
```

[116]:

	ExaminationGrade
1	C+
2	A
3	B

[117]:
```
#
# Create final XGrade DataFrame
#
var_ict1110_student_performance_dataframe_transformed_examinationGrade  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["ExaminationGrade"])
```

[118]:
```
#
# Inspect encoded column
var_ict1110_student_performance_dataframe_transformed_examinationGrade.head(3)
```

[118]:

	A	B	B+	C	C+	D	D+
1	0	0	0	0	1	0	0
2	1	0	0	0	0	0	0
3	0	1	0	0	0	0	0

XGPA—Categorical to Normal Encoding

[119]:
```
var_ict1110_student_performance_dataframe_transformed[["ExaminationGPA"]].head(3)
```

[119]:

	ExaminationGPA
1	2.37
2	4.00
3	3.00

[120]:
```
#
# Create final XGrade DataFrame
#
var_ict1110_student_performance_dataframe_transformed_examinationGPA  = pd.
 ↪get_dummies(var_ict1110_student_performance_dataframe_transformed["ExaminationGPA"])
```

[121]:
```
#
# Inspect encoded column
var_ict1110_student_performance_dataframe_transformed_examinationGPA.head(3)
```

[121]:

	0.00	1.50	2.37	3.00	3.50	4.00
1	0	0	1	0	0	0
2	0	0	0	0	0	1
3	0	0	0	1	0	0

Data Transformation Output

- Once data has been transformed, X input features are used to feed the estimators and, optionally, labels are used as output labels for training estimators that fall within the class for supervised machine learning

Usage Example: Predicting Examination Grade Using Accommodation Status

[122]:
```
# X input variables (independent variable)
var_ict1110_student_performance_dataframe_transformed_accommodated.head(2)
```

[122]:

	No	Yes
1	1	0
2	0	1

[123]:
```
# Y output variables (depedent variable)
var_ict1110_student_performance_dataframe_transformed_examinationGPA.head(5)
```

[123]:

	0.00	1.50	2.37	3.00	3.50	4.00
1	0	0	1	0	0	0
2	0	0	0	0	0	1
3	0	0	0	1	0	0
4	0	0	0	1	0	0
5	0	0	1	0	0	0

Save Pipeline

Sample Independent and Dependent Variables

Accommodation Status -> Test #1 Status

[124]:
```
#

#
# Independent variable
var_ict1110_student_performance_dataframe_transformed_accommodated.head(2)
len(var_ict1110_student_performance_dataframe_transformed_accommodated)
```

```
#
# Dependent variable
# Examination Status
var_ict1110_student_performance_dataframe_transformed_examinationStatus.head(2)
len(var_ict1110_student_performance_dataframe_transformed_examinationStatus)

#
#
```

[124]:

	No	Yes
1	1	0
2	0	1

[124]: 63

[124]:

	Fail	Pass
1	0	1
2	0	1

[124]: 63

[125]:
```
#
# Merge independent variables with dependent variables
# NOTE: this could be avoided by transforming the original dataset
#
var_ict1110_student_performance_accommodated_examination_model_input = pd.concat(
    [
        var_ict1110_student_performance_dataframe_transformed_accommodated,
        var_ict1110_student_performance_dataframe_transformed_examinationStatus
    ],
    axis="columns"
)
```

[126]:
```
var_ict1110_student_performance_accommodated_examination_model_input.head(3)
```

[126]:

	No	Yes	Fail	Pass
1	1	0	0	1
2	0	1	0	1
3	1	0	0	1

[127]:
```
# Save the merged example DataFrame
joblib.dump(var_ict1110_student_performance_accommodated_examination_model_input,␣
↪"var_ict1110_student_performance_accommodated_examination_model_input.pkl")
```

[127]:
```
['var_ict1110_student_performance_accommodated_examination_model_input.pkl']
```

[ ]:
```

```