	COLLEGE OF COMPUTING AND INFORMATION SCIENCES		
	Final-Term Assessment Fall 2020 Semester		
Class Id		**Course Title**	Introduction of Data Science
Program	BSCS	**Campus / Shift**	Main Campus / Morning
Date	12-12-2020	**Total Points**	85
Duration	03 hours	**Faculty Name**	Affan Alim
Student Id	9389	**Student Name**	Muhammad ibraheem

Instructions for Online submission

- Filling out Student-ID and Student-Name on the exam header is mandatory.
- Do not remove or change any part of the exam header or question paper.
- Write down your answers in the given space or at the end of the exam paper with the proper title "Answer for Question# _ _".
- Answers should be formatted correctly (font size, alignment, etc.)
- Handwritten text or image should be on A4 size page with clear visibility of contents.
- Only PDF format is accepted (Student are advised to install necessary software)
- In the case of CHEATING, COPIED material or any unfair means would result in negative marking or ZERO.
- A mandatory recorded viva session will be conducted to ascertain the quality of answer scripts where deemed necessary.

Caution: Duration to perform Final-Term Assessment is **03 hours only**. Extra 01 hours are given to cater to all kinds of odds in the submission of Answer-sheet. **Therefore, if you failed to upload the answer sheet on LMS (in PDF format) within 04 hours limit, you would be considered as ABSENT/FAILED.**

Instruction of Paper:

- Attempt all parts of the same question in the given order.
- Attempt all questions on the answer sheet.
- You're not allowed to assume anything. Strictly stick to the mentioned requirements.
- The sequence of your answer will be code, description, and output
- You may download datasets from https://drive.google.com/drive/folders/1wuns7AMQeanoJUNKG5HbWed46choyXZH?usp=sharing

[Problem-1, points: 25] titanic.csv

Consider the given dataset and answer the following questions with a one-line description. ***Without a description of each of your answer will not be marked.***

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C
10	11	1	3	Sandstrom, Miss. Marguerite Rut	female	4.0	1	1	PP 9549	16.7000	G6	S
11	12	1	1	Bonnell, Miss. Elizabeth	female	58.0	0	0	113783	26.5500	C103	S
12	13	0	3	Saundercock, Mr. William Henry	male	20.0	0	0	A/5. 2151	8.0500	NaN	S
13	14	0	3	Andersson, Mr. Anders Johan	male	39.0	1	5	347082	31.2750	NaN	S
14	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14.0	0	0	350406	7.8542	NaN	S

(i) Which attribute(s) have ordinal feature

Q1 part(i) Which attribute(s) have ordinal feature ¶

Ordinal Feature means those verable who have meaningful order in our dataset the ordinal feature ara

```
1=Ticket,
2=Pclass,
3=Embarked
```

(ii) Write the code to extract the title (Mr., Miss., Master, etc.) from the name attribute and add these titles into a new attribute named "Title".

Q1(ii)

Write the code to extract the title (Mr., Miss., Master, etc.) from the name attribute and add these titles into a new attribute named "Title".

i use split fun for extraction title from all data then after extracting i add Title attribute in our dataset

```
In [4]: Title=[x.split(' ')[1] for x in titanic.Name]
        titanic['Title']=Title
        titanic.head(10)
```

Out[4]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	Mr.
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	Mrs.
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	Miss.
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S	Mrs.
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S	Mr.
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q	Mr.
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E46	S	Mr.
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S	Master.
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	NaN	S	Mrs.
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	NaN	C	Mrs.

(iii) Write the code to fill the missing values in the Age attribute group by Title using the median property.

(iv) Write a code to remove those rows (instances) where Embarked has NaN values.

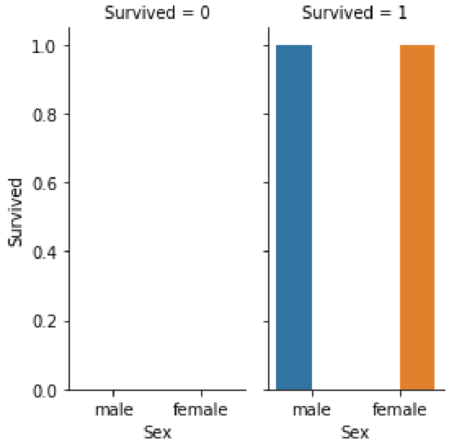
Q1 (iv) Write a code to remove those rows (instances) where Embarked has NaN values.

First check NaN values are exit or not,if exit then drop

```
In [14]: titanic.isnull().sum()

Out[14]: PassengerId      0
         Survived         0
         Pclass           0
         Name             0
         Sex              0
         Age            177
         SibSp            0
         Parch            0
         Ticket           0
         Fare             0
         Cabin          687
         Embarked         2
         dtype: int64

In [15]: titanicDF = titanic.dropna(how='any', subset=['Embarked'])

In [17]: titanicDF.isnull().sum()

Out[17]: PassengerId      0
         Survived         0
         Pclass           0
         Name             0
         Sex              0
         Age            177
         SibSp            0
         Parch            0
         Ticket           0
         Fare             0
         Cabin          687
         Embarked         0
         dtype: int64
```

(v) Write a code to draw a bar plot using “survived” and “Sex” attributes

```
.7]: # i use seaborn because i have better undestanding about sns
     titanicbarplot = sns.catplot(x="Sex", y="Survived",hue="Sex", col="Survived",
                     data=titanic, kind="bar",
                     height=4, aspect=.5);
     #titanic
```



[Problem-2, points: 25] weather_data.csv

Consider the following dataset and answer the following questions. ***Without a description of your answer will not be marked.***

	day	temperature	windspeed	event
0	1/1/2017	32	6us	Rain
1	1/4/2017	-9999	9	Sunny
2	1/5/2017	28	-7777	Snow
3	1/6/2017	-9999	7	NaN
4	1/7/2017	32 #	-7777	Rain
5	1/8/2017	-9999	-7777	Sunny
6	1/9/2017	-9999	-7777	NaN
7	1/10/2017	34FA	8yyy	Cloudy
8	1/11/2017	40	12	Sunny

(i) As we studied very large size values in any attribute of data should be considered a missing value or outlier. These values are filled with suitable NaN or any suitable values. Write a single line code to convert the -9999 and -7777 values into NaN.

Q2 (i) Write a single line code to convert the -9999 and -7777 values into NaN.

If any data show very high value then we use replace function for converting the high values into Nan.

```
In [27]: weather.head(10)
```

Out[27]:

	day	temperature	windspeed	event
0	1/1/2017	32	6us	Rain
1	1/4/2017	-9999	9	Sunny
2	1/5/2017	28	-7777	Snow
3	1/6/2017	-9999	7	NaN
4	1/7/2017	32 #	-7777	Rain
5	1/8/2017	-9999	-7777	Sunny
6	1/9/2017	-9999	-7777	NaN
7	1/10/2017	34FA	8yyy	Cloudy
8	1/11/2017	40	12	Sunny

```
In [28]: weather.replace(['-9999','-7777'],np.NaN)
```

Out[28]:

	day	temperature	windspeed	event
0	1/1/2017	32	6us	Rain
1	1/4/2017	NaN	9	Sunny
2	1/5/2017	28	NaN	Snow
3	1/6/2017	NaN	7	NaN
4	1/7/2017	32 #	NaN	Rain
5	1/8/2017	NaN	NaN	Sunny
6	1/9/2017	NaN	NaN	NaN
7	1/10/2017	34FA	8yyy	Cloudy
8	1/11/2017	40	12	Sunny

(ii) There are some unwanted characters in temperature and windspeed attributes like FA, yyy, us, etc. Write a code to remove these unnecessary characters from temperature and windspeed without changes the original values.

Q2(ii)(ii) There are some unwanted characters in temperature and windspeed attributes like FA, yyy, us, etc.

i use specific column replace fun if i use without column fun then this line remove all text values

```
31]: weather.replace({'temperature':'[A-Za-z]','windspeed':'[A-Za-z]'},'',regex=True)
```

31]:

	day	temperature	windspeed	event
0	1/1/2017	32	6	Rain
1	1/4/2017	-9999	9	Sunny
2	1/5/2017	28	-7777	Snow
3	1/6/2017	-9999	7	NaN
4	1/7/2017	32 #	-7777	Rain
5	1/8/2017	-9999	-7777	Sunny
6	1/9/2017	-9999	-7777	NaN
7	1/10/2017	34	8	Cloudy
8	1/11/2017	40	12	Sunny

(iii) Currently, the day attribute is an object type. Write a code to convert the "day" attribute type to date time. After conversion into date-time data type, covert day as an index of this dataset

Q2 (iii) Write a code to convert the "day" attribute type to date time. After conversion into date-time data type, covert day as an index of this dataset ¶

Day use for indexing Therefore is convert date in popper formate for converting in indexing

First we change datatype of day column then we convert this column into datetime index

In [12]:
```
weatherNew['day']=pd.to_datetime(weather['day'])
weatherNew.set_index('day',inplace=True)
weatherNew
```

Out[12]:

day	temperature	windspeed	event
2017-01-01	32	6	Rain
2017-01-04	-9999	9	Sunny
2017-01-05	28	-7777	Snow
2017-01-06	-9999	7	NaN
2017-01-07	32 #	-7777	Rain
2017-01-08	-9999	-7777	Sunny
2017-01-09	-9999	-7777	NaN
2017-01-10	34	8	Cloudy
2017-01-11	40	12	Sunny

(iv) Fill the missing values of temperature and windspeed with a suitable and most appropriate method. (you should justify why you have chosen this method)

Q2(iv)Fill the missing values of temperature and windspeed

i use mean in Temperature column and median in Windspeed because both attribute have Numerical values so therefore we use any one in both of them (mean,median)

[21]:
```
weather=weather.fillna({'temperature':weather['temperature'].mean()})
weather=weather.fillna({'windspeed':weather['windspeed'].median()})
```

(v) Draw a bar plot concerning a day on the horizontal axis a temperature & windspeed in vertical

[Problem-3, points: 20] shooting.csv

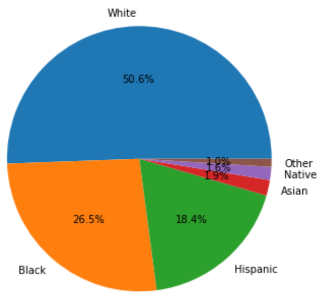
Consider the dataset given below and answer the following questions. ***Without a description of your answer will not be marked.***

(i) Write a code for drawing a pi-chart of an attribute "race" percentage-wise.

Q3(i)drawing a pi-chart of an attribute "race" percentage-wise

In [5]:
```
valueCounts=shooting.race.value_counts()
values=valueCounts.values
labels=valueCounts.index
autopct='%1.1f%%'
plt.figure(figsize=(12,6))
plt.pie(values,labels=labels,autopct=autopct)
plt.show
```

Out[5]: <function matplotlib.pyplot.show(*args, **kw)>

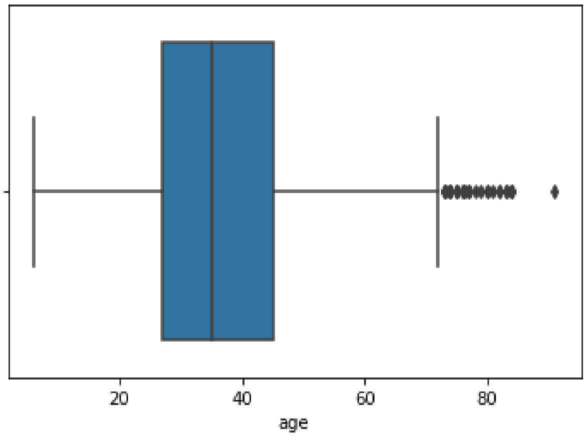


(ii) Write a code for drawing a box-plot for finding the outliers

```
# i show olny one age attribute outliers
```

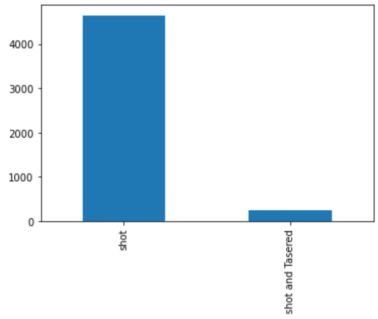
```
sns.boxplot(x=shooting['age'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x141b2214fd0>
```



(iii) Write a code for drawing a bar chart comparison between manner_of_death and races.

```
x = shooting["manner_of_death"].value_counts().plot(kind='bar')
plt.show()
```



(iv) Write a code for dummy encoding in “manner_of_death” values.

```
#Q3(iv)(iv)=Write a code for dummy encoding in “manner_of_death” values
#shooting.info()
```

```
Counts=shooting['manner_of_death'].value_counts()
Counts
```

```
shot                4647
shot and Tasered     248
Name: manner_of_death, dtype: int64
```

```
newdata=pd.get_dummies(shooting,columns=['manner_of_death'],drop_first=True,prefix='MOD')
newdata
```

id	name	date	armed	age	gender	race	city	state	signs_of_mental_illness	threat_level	flee	body_camera	arms_category	MOD_shot and Tasered
3	Tim Elliot	2015-01-02	gun	53.0	M	Asian	Shelton	WA	True	attack	Not fleeing	False	Guns	0
4	Lewis Lee Lembke	2015-01-02	gun	47.0	M	White	Aloha	OR	False	attack	Not fleeing	False	Guns	0
5	John Paul Quintero	2015-01-03	unarmed	23.0	M	Hispanic	Wichita	KS	False	other	Not fleeing	False	Unarmed	1
8	Matthew Hoffman	2015-01-04	toy weapon	32.0	M	White	San Francisco	CA	True	attack	Not fleeing	False	Other unusual objects	0
	Michael	2015-									Not			

	id	name	date	manner_of_death	armed	age	gender	race	city	state	signs_of_mental_illness	threat_level	flee	body_camera	arms_
0	3	Tim Elliot	2015-01-02	shot	gun	53.0	M	Asian	Shelton	WA	True	attack	Not fleeing	False	
1	4	Lewis Lee Lembke	2015-01-02	shot	gun	47.0	M	White	Aloha	OR	False	attack	Not fleeing	False	
2	5	John Paul Quintero	2015-01-03	shot and Tasered	unarmed	23.0	M	Hispanic	Wichita	KS	False	other	Not fleeing	False	
3	8	Matthew Hoffman	2015-01-04	shot	toy weapon	32.0	M	White	San Francisco	CA	True	attack	Not fleeing	False	Othe
4	9	Michael Rodriguez	2015-01-04	shot	nail gun	39.0	M	Hispanic	Evans	CO	False	attack	Not fleeing	False	Piercir
5	11	Kenneth Joe Brown	2015-01-04	shot	gun	18.0	M	White	Guthrie	OK	False	attack	Not fleeing	False	
6	13	Kenneth Arnold Buck	2015-01-05	shot	gun	22.0	M	Hispanic	Chandler	AZ	False	attack	Car	False	
7	15	Brock Nichols	2015-01-06	shot	gun	35.0	M	White	Assaria	KS	False	attack	Not fleeing	False	
8	16	Autumn Steele	2015-01-06	shot	unarmed	34.0	F	White	Burlington	IA	False	other	Not fleeing	True	
9	17	Leslie Sapp III	2015-01-06	shot	toy weapon	47.0	M	Black	Knoxville	PA	False	attack	Not fleeing	False	Othe

<u>[Problem-4, points: 15] (don’t need to clean, you may use normalize and features reduction method)</u>

Use the given dataset (heart.csv) for classification, apply at least two machine learning models (one tree-based and other non-tree-based) and fill the following table.

i. Show the confusion matrix
Confusion matrix show below code

ii. Fill the table below with code and output from where you will fill the table (only filled table will be marked)

Code show below screenshoot

iii. What ML method is more efficient when data is normalized?
Ans:Yes Decision Tree ML method is more efficient for Normalized data.

iv. Which ML method can be handled the NaN without filling?
Decision Tree

Sno	ML Model	Train-Test ratio	Test Accuracy	Sensitivity	Specificity	AUC	Precision	Classification error
1	Decisio nTree	70/30	0.7704 9	0.7812 5	0.7586 20		0 0.76 1 0.78	
2	Logistic Regression	70/30	0.8196 7	0.9354 83	0.7		0 0.9 1 0.7	

Logistic Regressin gave batter result as compate to DT

Decision Tree

```
from sklearn.model_selection import train_test_split
```

```
n [ ]:
```

```
n [2]: heart=pd.read_csv('heartFT.csv')
       heart.head()
```

ut[2]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1

```
n [3]: InputTrain_x=heart.drop('target',axis=1)
       InputTarget_y=heart['target']

n [4]: train_x,test_x,train_y,test_y=train_test_split(InputTrain_x,InputTarget_y, test_size=0.2)

n [5]: from sklearn import tree
       model=tree.DecisionTreeClassifier()
       model.fit(train_x,train_y)

ut[5]: DecisionTreeClassifier()

n [6]: DT_pred=model.predict(test_x)

n [7]: accu=accuracy_score(test_y,DT_pred)
       print(accu)

       0.7704918032786885

In [9]: from sklearn.metrics import classification_report, confusion_matrix
        results=confusion_matrix(test_y,DT_pred)

:n [10]: print(results)
         tn,fp,fn,tp=confusion_matrix(test_y,DT_pred).ravel()
         print('tn',tn)
         print('fp',fp)
         print('fn',fn)
         print('tp',tp)
         print("")
         se=tp/(tp+fn)
         sp=tn/(tn+fp)
         print('sensitivity=',se)
         print('specifity=',sp)
         print("")
         print(classification_report(test_y,DT_pred))

         [[22  7]
          [ 7 25]]
         tn 22
         fp 7
         fn 7
         tp 25

         sensitivity= 0.78125
         specifity= 0.7586206896551724

                       precision    recall  f1-score   support

                    0       0.76      0.76      0.76        29
                    1       0.78      0.78      0.78        32

             accuracy                           0.77        61
            macro avg       0.77      0.77      0.77        61
         weighted avg       0.77      0.77      0.77        61
```

Logistic Regression

```
]: import pandas as pd
   from sklearn.linear_model import LogisticRegression
   from sklearn.metrics import accuracy_score
   import matplotlib.pyplot as plt
   from sklearn.model_selection import train_test_split
```

```
]: data=pd.read_csv('heartFT.csv')
   print(data.shape)
```

```
(303, 14)
```

```
]: #print(data)
   #data.head(10)
   #data
```

```
]: x=data.drop('target',axis=1)
   y=data['target']
```

```
]: train_x,test_x,train_y,test_y=train_test_split(x,y, test_size=0.2)
```

```
]:
```

```
]: #Model
   model=LogisticRegression()
   model.fit(train_x,train_y)
```

```
Out[6]: LogisticRegression()
```

```
In [7]: y_predicted=model.predict(test_x)
        #print(y_predicted)
```

```
In [8]: test_Acc=accuracy_score(test_y,y_predicted)
        print("test accuracy=",test_Acc)
```

```
test accuracy= 0.819672131147541
```

```
In [9]: ## confusion_matrix_Logistic_Regression
```

```
In [10]: from sklearn.metrics import classification_report, confusion_matrix
         results=confusion_matrix(test_y,y_predicted)
```

```
In [11]: print(results)
         tn,fp,fn,tp=confusion_matrix(test_y,y_predicted).ravel()
         print('tn',tn)
         print('fp',fp)
         print('fn',fn)
         print('tp',tp)
         print("")
         se=tp/(tp+fn)
         sp=tn/(tn+fp)
         print('sensitivity=',se)
         print('specifity=',sp)
         print("")
         print(classification_report(test_y,y_predicted))
```

```
[[21  9]
 [ 2 29]]
tn 21
fp 9
fn 2
tp 29

sensitivity= 0.9354838709677419
specifity= 0.7

              precision    recall  f1-score   support

           0       0.91      0.70      0.79        30
           1       0.76      0.94      0.84        31

    accuracy                           0.82        61
   macro avg       0.84      0.82      0.82        61
weighted avg       0.84      0.82      0.82        61
```