PART A - Spreadsheet (Excel)

Experiment - 1

Aim: Perform Conditional formatting, if, countif, sumif, average and concat functions on a dataset in excel

CONDITIONAL FORMATTING:

Conditional formatting can help make patterns and trends in your data more apparent. To use it, you create rules that determine the format of cells based on their values.

IF function:

It allows you to make logical comparisons between a value and what you expect

Syntax:

=IF(logical test, value if true, value if false)

COUNTIF function:

Use countif function to count the number of cells that meet a criteria.

Syntax:

=COUNTIF(range, criteria)

SUM IF function:
Use SUMIF function to sum the value in a range that meet the criteria you specify.
Syntax:
=SUMIF (range, criteria, sum_range)

AVERAGE function:
Returns the average (arithmetic mean) of the arguments. F
Syntax:
=AVERAGE ([number 1], [number 2], ...)

CONCAT function:
The concat function combines the text from multiple ranges or strings but it does not provide delimeter or IgnorEmpty arguments
Syntax:
CONCAT ([text 1], [text 2]....)

Experiment-2

Aim: To perform INDEX, MATCH, UNIQUE, IFS, COUNTIFS, SUMIFS, AVERAGEIFS on a dataset in excel.

INDEX function:

Returns the value of an element in a table or an array, selected by the row and column number indices.

Syntax:

=INDEX (array, row_num, [col_num])

MATCH function:

The MATCH function searches for a specified item in a range of cells, and then returns the relative position of that item in the range.

Syntax:

=MATCH (lookup_value, lookup_array, [match_type])

UNIQUE function:

The UNIQUE function returns a list of unique values in a list or range.

Syntax:

=UNIQUE (range / list)

IFS function:

The IFS function checks whether one or more conditions are met and returns a value that corresponds to the first TRUE condition.

Syntax:

=IFS (logical_test1, value_if_true1, ...)

COUNTIFS function:

COUNTIFS function applies criteria to cells across multiple ranges and counts the number of times all criteria are met.

Syntax:

=COUNTIFS (criteria range1, criteria1, [criteria range2], [criteria2] ...)

SUMIFS function:

The SUMIFS function adds all of its arguments that meet multiple criteria.

Syntax:

=SUMIFS (sum_range, criteria_range1, criteria1, ...)

AVERAGEIFS function:
Returns the average (arithmetic mean) of all cells that meet multiple criteria.

Syntax:
=AVERAGEIFS (average_range, criteria range1, criteria1, ...)

Experiment 3

Aim: Perform VLOOKUP, HLOOKUP, XLOOKUP, COUNT, COUNTA functions on a dataset in excel.

VLOOKUP function:

Use VLOOKUP when you want to find things in a table or a range by row.

Syntax:

=VLOOKUP(lookup_value, table_array, col_index_num, [range_lookup])

HLOOKUP function:

Searches for a value in the top row of a table or an array of values, and then returns a value in the same column from a row you specify in the table or array.

Syntax:

=HLOOKUP(lookup_value, table_array, row_index, [range_lookup])

XLOOKUP function:

The XLOOKUP function searches a range or an array and then returns the item corresponding to the first match it finds.

Syntax:

=XLOOKUP(lookupvalue, lookuparray, returnarray, [if_not_found], [match_mode], [searchmode])

COUNT :

The COUNT function counts the number of cells that contain numbers, and counts numbers within the list of arguments.

Syntax :

COUNT (value1, value2, . . .)

COUNTA function :

The COUNTA function counts the number of cells that are not empty in a range.

Syntax :

COUNTA (value 1, [value 2], . . .)

Experiment 4.

Aim: Perform LEFT, MID, RIGHT, LEN, SUBSTITUTE, SEARCH, ISNUMBER on a dataset in Excel.

LEFT function:

LEFT returns the first character or characters in a text string, based on the number of characters you specify.

Syntax:

=LEFT(text, [num_chars])

MID function:

MID returns a specific number of characters from a text string, starting at the position you specify, based on the number of characters you specify.

Syntax:

MID (text, startnum, num_chars)

RIGHT function:

Right returns the last character or characters in a text, based on the number of characters you specify

Syntax:

=RIGHT (text, [num_chars])

LEN functions:
LEN returns the number of characters in a text string:
Syntax:
LEN (text)

SUBSTITUTE function:
Substitutes new_text from old_text in a string.
Syntax:
=SUBSTITUTE (text, old_text, new_text, [instance_num])

SEARCH function:
The search function locates one text string within a second text string and return the number of the starting position of the first text string from the first character of the second text string.
Syntax:
=SEARCH (find_text, within_text, [start_num])

Date :..... Expt./ Programme No :..... Page No:.....

ISNUMBER function:

Returns TRUE if value refers to a number

Syntax:

=ISNUMBER (value)

Experiment 5:

Aim: Perform TODAY, NOW, YEAR, MONTH, NETWORKDAYS, EOMONTH functions a dataset in Excel.

TODAY function:

Returns the serial number of the current date.

Syntax:

=TODAY()

NOW function:

Returns the serial number of the current date and time.

Syntax:

NOW()

YEAR function:

Returns the corresponding year to a date. The year is returned as an integer in the range 1900-9999.

Syntax:

YEAR(serial_number)

MONTH function:
Returns the month of date represented by a serial number. The month is given as an integer, ranging from 1 (January) to 12 (December).

Syntax:
MONTH (serial_number)

NETWORKDAYS function:
Returns the number of whole working days between start_date and end_date. Working days exclude weekends and any dates identified in holidays.
Syntax:
=NETWORKDAYS (start_date, end_date, [holidays])

EOMONTH function:
Returns the serial number for the last day of the month that is the indicated number of months before or after start_date
Syntax:
=EOMONTH (startdate, months)

Experiment 6:

Aim: Perform OFFSET, CHOOSE, LET, MAX, SORT, SORTBY, RANK on a dataset in Excel.

OFFSET:

Returns a ~~value~~ reference to range that is a specified number of rows and columns from a cell or range of cells.

Syntax:

OFFSET(reference, rows, cols, [height], [width])

CHOOSE:

Uses index_num to return a value from the list of value arguments.

Syntax:

CHOOSE (index_num, value1, [value2], ...)

LET:

The LET function assigns names to calculation results.

Syntax:

LET (name1, value1, name2, value2, ...)

Date :..... Expt./ Programme No :..... Page No:.....

MAX :

Returns the largest value in a set of values.

Syntax:

MAX (num1, num2,...)

SORT :

The SORT function sorts the contents of a range or array.

Syntax:

=SORT (array, [sort_index], [sort_order], [by_col])

SORTBY :

The SORTBY function sorts the contents of a range or array based on the values in a corresponding range or array.

Syntax:

=SORTBY (array, by_array1, [sort order], [by_array2], [sort_order2], ...)

Date :..... Expt./ Programme No :..... Page No:.....

RANK:

Returns the rank of a number in a list of numbers.

Syntax:

=RANK(number, ref, [order])

Experiment 7:

Aim: Perform FILTER, FREQUENCY, SEQUENCE, RANDARRAY, IFERROR on a dataset in Excel.

FILTER:

The filter function allows you to filter a range of data based on the criteria you define.

Syntax:

=FILTER(array, include, [if-empty])

FREQUENCY:

The frequency function calculates how often values occurs with a range of values.

Syntax:

=FREQUENCY (data_array, bins_array)

SEQUENCE:

The SEQUENCE function allows you to generate a list of sequential numbers in an array.

Syntax:

=SEQUENCE (rows, columns, start, step)

Date :..... Expt./ Programme No :..... Page No:.....

RANDARRAY:

Returns an array of Random Numbers.

Syntax:

=RAND ARRAY ([rows], [columns], [min], [max], [whole-number])

IFERROR:

IFERROR function is used to trap & handle errors in a formula.

Syntax:

IFFERROR(value, value_if_error)

Experiment 8:

Aim: Use PIVOT TABLE, WHATIF ANALYSIS, DATA VALIDATION, SUBTOTALS WITH RANGE on a dataset in Excel.

PIVOT TABLE:

A pivot table is a powerful tool to calculate, summarize and analyze data that lets you see comparisions, patterns, and trends in your data.

WHATIF ANALYSIS:

By using whatif analysis tools in Excel, you can use several different sets of values in one or more formulas to explore all the various results.

Date :..... Expt./ Programme No :..... Page No:.....

DATA VALIDATION:

Use data validation to restrict the type of data on the values that users enter into a cell, like drop down list.

SUBTOTALS WITH RANGES:

Returns a subtotal in a list or database.
It is generally easier to create a list with subtotals by using the subtotal command in outline group on the Data Tab in Excel.

PART-B : Data Analysis using Python

Program 1 : Probability

a. Calculating the simple probabilities.

```
import numpy as np
import pandas as pd

df = pd.read_csv("Book1.csv")
print(df)

pen = df['Book'].tolist()
print(pen)
#Mean
print ("Mean:", np.mean(pen))
#Variance
print ('Variance:', np.var(pen))
#Standard deviation
print ('Standard Deviation:', np.std(pen))
```

6. Applications of Probability distributions in real life problems.

```
# Binomial distributions
from scipy.stats import binom
n=6
p=0.6
r_value= list(range(n+1))
mean,var=binom.stats(n,p)
dist=[binom.pmf(r,n,p) for r in r_values]
print("r\tp(r)")
for i in range(n+1):
    print(str(r_values[i])+"\t"+str(dist[i]))
print("mean="+str(mean))
print("variance="+str(var))
```

```
#plotting graph for binomial distributions.
from scipy.stats import binom
from matplotlib.pyplot as plt.

n=6
p=0.6
r_values= list(range(n+1))
print(r_values)
dist=[binom.pmf(r,n,p) for r in r_values]
plt.bar(r_value,dist)
plt.show()
```

Date :..... Expt./ Programme No :..... Page No:.....

#Normal Distribution.

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm
import statistics

x_axis = np.arrange(-20,20, 0.01)
mean = statistics.mean(x_axis)
sd = statistics.stdev(x_axis)
plt.plot(x_axis, norm.pdf(x_axis, mean, sd))
plt.show
```

```
# Poisson Distribution
from numpy import random
import matplotlib.pyplot as plt
import seaborn as sns

sns.distplot(random.poisson(lam=2, size=1000),
             kde=False)
plt.show()


#.
from scipy.stats import poisson
import seaborn as sp
data_binom = poisson.rvs(mu=4, size=10000)
ax=sb.distplot(data_binom,
               kde=True,
               color='green',
               hist_kws={"linewidth":25,'alpha':1})
ax.set(xlabel='Poisson', ylabel='Frequency')
```

Program 2:

Test of significance

a. T-Test : one sample, two independent samples and paired.

```
# To demonstrate one sample T-test

import scipy.stats as stats
import pandas as pd

data=pd.read_csv('area.csv')
t_statistics, pvalue = stats.ttest_1samp(a=data,
popmean=5000)
print(t_statistic, p_value)

# To demonstrate two sample T-test
datagroup1 = np.array([14,15,15,16,13,8,14,
                       17,16,14,19,20,21,15,
                       15,16,16,13,14,12])

data_group2 = np.array([15,17,14,17,14,18,12,
                        19,19,14,17,22,24,16,
                        13,16,13,18,15,13])

stats.ttest_ind(a=data_group1, b=data_group2,
                equal_var=True)
```

b. ANOVA & Chi-Square Test

#ANOVA

```
from scipy.stats import f_oneway

performance1= [89,89,88,78,79]
performance2=[93,92,94,89,88]
performance3=[89,88,89,93,90]
performance4=[81,78,81,92,82]

f_oneway(performance1,performance2,performance3,
         performance4)
```

CHI-SQUARE TEST

```
from scipy.stats import chi2_contingency

data =[[207,282,241],[234,242,232]]
stat,p,dof,expected = chi2_contingency(data)

alpha= 0.05
print("pvalue is " + str(p))
if p<=alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (H0 holds true)')
```

Program 3:
Correlation and regression analysis
a. Scattered diagram, calculating of correlation coefficient.

```
# Scattered diagram.
import numpy
import matplotlib.pyplot as plt

x= numpy.random.normal(5.0, 1.0, 1000)
y= numpy.random.normal(10.0, 2.0, 1000)

plt.scatter(x, y)
plt.show()

# Correlation coefficient.
import math
def correlationCoefficient(X, Y, n):
        sum_X = 0
        sum_Y = 0
        sum_XY = 0
        squareSum_X = 0
        squareSum_Y = 0
        i = 0
        while i<n:
            sum_X = sum_X + X[i]
            sum_Y = sum_Y + Y[i]
            sum_XY = sum_XY + X[i] * Y[i]
            squareSum_X = squareSum_X + X[i] * X[i]
            squaresum_Y = squareSum_Y + Y[i] * Y[i]
            i = i + 1
    corr = (float) (n*sumXY - sumX*sumY) /(float)
        (math.sqrt((n* squareSum_X - sum_X*sum_X)
        *(n*squareSum_Y - sum_Y*sum_Y)
```

~~return~~ return corr

#Driver function

X=[15,18,21,24,27]

Y=[25,25,27,31,32]

n=len(X)

print('{0:.6f}'.format(correlationCoefficient(X,Y,n)))

b. Linear Regression : fitting, testing model adequacy and prediction (simple and multiple)

~~a. Fitting of~~

```
import numpy as np
import matplotlib.pyplot as plt

def estimate_coef(x, y):
    n = np.size(x)
    m_x = np.mean(x)
    m_y = np.mean(y)
    SS_xy = np.sum(y*x) - n*m_y*m_x #crossdeviation
    SS_xx = np.sum(x*x) - n*m_x*m_x #deviationabout x
    b_1 = SS_xy / SS_xx
    b_0 = m_y - b_1*m_x
    return (b_0, b_1)

def plot_regression_line(x, y, b):
    plt.scatter(x, y, color = "m", marker = "o", s = 30)
    y_pred = b[0] + b[1]*x
    plt.plot(x, y_pred, color = "g")
    plt.xlabel('x')
    plt.ylabel('y')
    plt.show()

def main():
    x = np.array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
    y = np.array([1, 3, 2, 5, 7, 8, 8, 9, 10, 12])
    b = estimate_coef(x, y)
    print("Estimated coefficients: \nb_0 = {} \
          \nb_1 = {}".format(b[0], b[1]))
    plot_regression_line(x, y, b)
if __name__ == "__main__":
    main()
```

e. Fitting logistic regression

```
import pandas as pd
dataset = pd.read_csv("User_Data1.csv")

x = dataset.iloc[:,[2,3]].values

y = dataset.iloc[:,4].values

print(dataset)

#Splitting The dataset: Train and Test dataset

from sklearn.model_selection import train_test_split
    x, y, test_size = 0.25, random_state = 0)

xtrain, xtest, ytrain, ytest = train_test_split(x, y,
  testsize = 0.25, random_state = 0)

from sklearn.preprocessing import StandardScaler
sc_x = StandardScaler()
xtrain = sc_x.fit_transform(xtrain)
xtest = sc_x.transform(xtest)

print(xtrain[0:10], :])
#Train the model
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(xtrain, ytrain)

y_pred = classifier.predict(xtest)
```

```
#Evaluation metrics:
from sklearn.metrics import confusion_matrix

cm = confusion_matrix (ytest, y_pred)
print ("Confusion matrix : \n", cm)

#visualizing the performance.
from matplotlib.colors import ListedColormap
import numpy as np
from matplotlib import pyplot as plt

X_set, y_set = xtest, ytest
X1, X2 = np.meshgrid (np.arange (start = X_set[:, 0].
                                  min() - 1,
                                  stop = X_set[:, 0].max()+1, step=0.01),
                      np.arange (start = X_set[:, 1].min()-1,
                      stop = X_set[:, 1].max()+1, step = 0.01))
plt.contourf(X1, X2, classifier.predict (
    np.array ([X1.ravel (), X2.ravel()]).T).
    reshape ( X1.shape), alpha=0.75, cmap = ListedColormap
        (('red', 'green')))
plt.xlim (X1.min(), X1.max())
plt.ylim (X2.min(), X2.max())

for i, j in enumerate (np.unique (y_set)):
    plt.scatter (X_set[y_set == j, 0], X_set[y_set == j, 1],
                 c = ListedColormap (('red', 'green')) (i),
                 label = j )
plt.title ('classifier (Test set)')
plt.xlabel ('Age')
plt.ylabel ('Estimated Salary')
plt.legend ()
plt.show()
```

PART - C : Power BI

Experiment 1:

Aim : To Create a report in Power BI

Step 1: Open Power BI Desktop

Step 2: In the 'Home' tab click 'Get Data' in 'Data' section and select text/csv file and chose the file

Step 3: Navigator window will open and then select the required tables.

Step 4: Click Load and wait for the table to load.

Step 5: In the 'visualizations' tab on the Right hand side select the line graph under 'Build visual'.

Step 6: Select the columns you want in the 'Data' tab on the Right hand side. This creates a line graph.

Step 7: Click on 'Card' in 'Build visual' in 'Visualizations' tab and select a column from 'Data' tab to display on the card.

Step 8: From 'Data' tab select any column to display on the report. The graphs and cards keep changing on which value you select on the displayed column.

Step 9: Click on 'publish' in the Home tab on in the 'share' section. Then save your report and select a destination.

Step 10: Then on the next window click on the link. 'Open <filename> in Power BI' to view the report.

Experiment 2:

Aim: to Edit a report

Step 1: Open any report that you have created.
Step 2: Click on the 'Edit' button on the status bar.
Step 3: Select the graphs you want to edit
Step 4: You can add columns from 'data' tab on Right hand side to the graph
Step 5: Format your visual using 'Format your visual' from 'visualizations' tab.
Step 6: Save the report using 'save' button which is the fourth button from the Right hand side on the status bar.

Experiment - 3:

Aim: To Create slicers in Power BI

Step 1: Open Power BI desktop.

Step 2: Select 'slicer visual' from 'build visual' in 'Visualization' tool in the Right hand side.

Step 3: Pick the category you need in the slicer

Step 4: ~~You~~ Go to 'Format you visual' option in ~~the~~ 'Visualizations' tool and under options select 'Tile' for button slicers.

Step 5: Go to 'Selection' and select single select to remove multiple selection.

Step 6: Create a new page.

Step 7: Select 'slicer visual' like in Step 2

Step 8: Select + more that one columns to add a hierarchy slicer.

Step 9: Create a new Page.

Step 10: Select 'Slicer visual' liken in Step 2:

Step 11: Select a column with neumeric values ~~to~~ and the Slicer will automatically can to a number slider slicer.

Step 12: Create a new page

Step 13: Slect 'Slicer Visual' like in Step2

Step 14: Select a column with 'Date' or any other Calendar values to create a timeline slicer. You can change ~~slide~~ the slider value or chose dates directly in this slicer.

Step 15: Select any slicer and select 'Sync Slicers' in 'view' tab and select the slicers you want to sync.

Experiment 4:

Aim: To Export a Report to Powerpoint

Step1: Open any existing report that you have created in any browser of choice.

Step 2: Click on the 'Export' button on the Status bar.

Step 3: Select 'PowerPoint' option ~~on~~ dropdown menu and click 'Embed live data'

Step 4: A window named 'Embed live data in Power Point' will open and click on ~~Step~~ 'Open in PowerPoint button'.

Step 6: Power Point will open ask ask to sign in to your Power BI account after that the report will be displayed on the powerpoint slide and it can be interacted with.