**Group\_9\_Task 4**:

Create a Spark Pipeline, Create a Pipeline for 1-3 tasks

**Approach:**

We have combined all the code for tasks 1-3 and then added pipelines for each stage. For task 1 we added the pipeline after the string indexer and then went ahead and used the knnimputer on the same data. For tasks 2 & 3 we have used a combined and used pipeline before the linear regression and its evaluate function to get RMSE and before the random forest regression and its RMSE and obtained the pipelined output predictions and the RMSE from the pipeline stage

**Dataset (Input and Output):** Similar to the task1 we have used the input dataset provided written it to our system and then used it as the input file. The cleaned and corrected data file is written in CSV format to the local from task1, which is the input for both task2 and task 3. The output of the first task pipeline would be the file written to local, the output for the task2 pipeline would be the predictions and the final output would be the RMSE values from task3.

* First we have imported all the required libraries from task1,2 and 3. Then we have inserted pipeline stages as mentioned above on the same code.
* **Reading data:** We utilized a data frame reader to read the data from the data source to load the path of the data specified for the specific question and to load the data frame in the same format as the tabular format.
* Once all the stages of the pipeline are ready, we started training the model. We created a pipeline object with all of the stages included in it. The fit() is used which would return the pipeline model object which is used to make a prediction of the edu. Then we use transform() function to make predictions on the testing data. We have used the same way for creating a pipeline for each task and finally getting the outputs for the specific task.

**Instructions to compile and run the program:**

**Imports Used:**

from pyspark.sql import SparkSession

from pyspark.ml.linalg import Vectors

from pyspark.ml.feature import Imputer

from pyspark.sql.types import IntegerType

from pyspark.sql.functions import col

from pyspark.sql import functions as F

from pyspark.ml.regression import LinearRegression

import pandas as pd

import numpy as np

from pyspark.ml.feature import StringIndexer

from pyspark.ml.feature import VectorAssembler

from pyspark.ml.classification import RandomForestClassifier

from pyspark.ml.evaluation import BinaryClassificationEvaluator

from pyspark.ml.evaluation import RegressionEvaluator

from itertools import chain

from pyspark.sql.functions import count, mean, when, lit, create\_map, regexp\_extract

from pyspark.ml.classification import RandomForestClassifier

from pyspark.ml.evaluation import MulticlassClassificationEvaluator

from pyspark.ml import Pipeline

from pyspark.ml.tuning import CrossValidator, ParamGridBuilder

from pyspark.ml.linalg import Vectors

from pyspark.ml.regression import LinearRegression

from pyspark.sql import SparkSession,DataFrame

from pyspark import SparkContext, SparkConf

from pyspark.sql import\*

from pyspark.ml import Pipeline

**Input:** As mentioned, we have used the input file of Task 1 as the input for Task 4. ‘‘/user/kaggle/kaggle\_data/marketing\_campaign.csv’’. The output from the task1 (Task\_4\_output\_1.csv) is the input for task 2and 3

**Output:** From task1 is the output file written to local(Task\_4\_output\_1.csv), for task2 it is the predictions for column ‘Edu’, for task3, it is the RMSE value

1. Open the cluster and login into it.

2. Upload the Group\_9\_Task\_4.py file into session storage(local or csh)

3. Command: spark-submit Group\_9\_Task\_4.py

**Discussion of Results:**

We have created pipelines for all the mentioned tasks above, we have used the pipeline model and fit() and transform() methods to create pipeline stages for all the three tasks. From the output generated for each task we have observed that they’re similar to the outputs generated by the individual tasks give.