Por una universidad de **excelencia** y **solidaria**

Universidad del Cauca

ISO 9001:2015 SC- CER 450832

IQNet: CO- SC-CER450832

PROYECTO FINAL
OPTIMIZACION DEL EFOQUE CLR MEDIANTE ALGORITMOS GENETICOS

Estudiante: Germán Homero Morán Figueroa
Profesor: Dra. Martha Eliana Mendoza Becerra

Optimización usado Metahuristicas

Maestría en Computación
Facultad de Ingeniería Electrónica y Telecomunicaciones – FIET

Por una universidad de excelencia y solidaria

Universidad del Cauca

CONTENIDO

Por una universidad de excelencia y solidaria

Universidad del Cauca

1. INTRODUCCIÓN

GENERALIDADES

- Importancia del cultivo del maíz
- Actualmente producción nacional solo logra cubrir 80 %, 20% restante se debe a la exportación.
- Federación Nacional de Cultivadores de Cereales y Leguminosas (FENALCE).
- El departamento de córdoba aporta el 11,4% de producción nacional, con cultivos de maíz tradicional y tecnificado.
- Cada vez que un agricultor realiza una siembra existe un evento único.

2. DEFINICIÓN DEL PROBLEMA

OBJETIVO DEL NEGOCIO.

DETERMINAR EL RENDIMIENTO DEL CULTIVO DE MAIZ TENIENDO EN CUENTA DIFERENTE VARIABLES QUE PUEDEN AFECTAR EL RENDIMEINTO

FUENTES DE INFROMACIÓN

VISTA MINABLE

	TIPO_SIEMBRA	SEM_TRATADAS	DIST_SURCOS	DIST_PLANTAS	COLOR_ENDOSPERMO	SEM_POR_SITIO	MATERIAL_GENETICO	CULT_ANT	RDT_AJUSTADO
0	Mecanizado	NO	0.8	0.2	Blanco	2	PIONEER 30F32	Algodon	4767.44
1	Mecanizado	SI	0.8	0.2	Blanco	1	DK 234	Maiz	4651.16
2	Mecanizado	NO	0.8	0.2	Blanco	1	PIONEER 30F32	Algodon	5180.23
3	Mecanizado	NO	0.8	0.2	Blanco	1	Otro	Algodon	4897.67
4	Mecanizado	NO	0.8	0.2	Blanco	1	Otro	Algodon	5302.33
5	Mecanizado	SI	0.8	0.2	Blanco	2	DK 234	Maiz	4958.14
6	Manual	SI	0.8	0.2	Blanco	3	Otro	Maiz	4426.74
7	Manual	SI	0.8	0.2	Blanco	3	PIONEER 30F32	Algodon	5790.70
8	Manual	SI	0.8	0.2	Blanco	2	DK 234	Algodon	5116.28
9	Mecanizado	NO	0.8	0.2	Amarillo	1	PIONEER 30F35	Frijol	4238.37

 # Variables: 95
 # Observaciones: 747 Registros

 # Variables Numéricas: 64
 # Variables categóricas: 31

Por una universidad de excelencia y solidaria

Universidad del Cauca

MODELADO

attribute	wei...
MATERIAL_GENETICO = DK 234 YGRR	1
MATERIAL_GENETICO = DK 1596	1
POSICION_PERFIL_RASTA = PLA0 CON ONDULACIONES	1
POSICION_PERFIL_RASTA = LADERA CONVEXA	1
POSICION_PERFIL_RASTA = LADERA CONCAVA	1
ESTRUCTURA_RASTA = MA1VA	1
OBSERVA_PLANTAS_PEQUENAS_RASTA = PLANTAS 0RMALES	1
OBSERVA_PLANTAS_PEQUENAS_RASTA = POCO AFECTADAS	1
OBSERVA_PLANTAS_PEQUENAS_RASTA = MUY AFECTADAS	1
drenaje_externo = LENTO	1
COLOR_ENDOSPERMO	1
DRENAJE	1
CAP_ENDURE_RASTA	1
POBLACION_20DIAS_AJT	1
ContEnfQui_Emer_Flor	1

SELECCIÓN DE ATRIBUTOS – 30

attribute	wei...
ContEnfQui_Emer_Flor	1
ContMalMec_Siem_Emer	1
ContPlaQui_Antes_Siem	1
ESPESOR_CAP_ENDURE_RASTA	1
Porc_Ar	1
Porc_PLASTICO	1

rapidminer

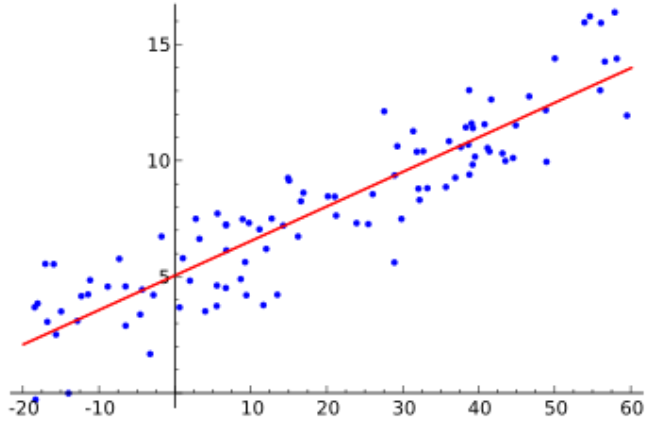
Por una universidad de excelencia y solidaria

Universidad del Cauca

MODELADO

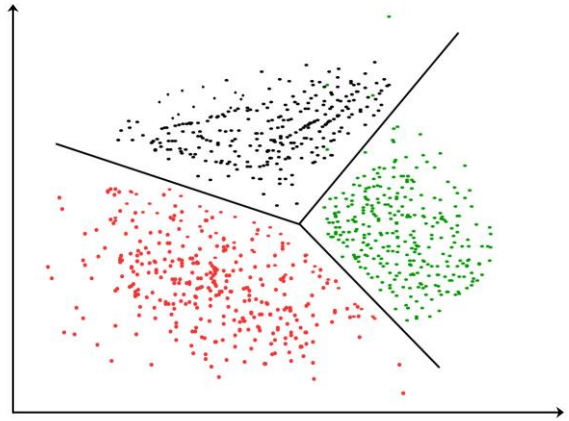
Selección de la técnica del Modelado
Algoritmos Supervisados:

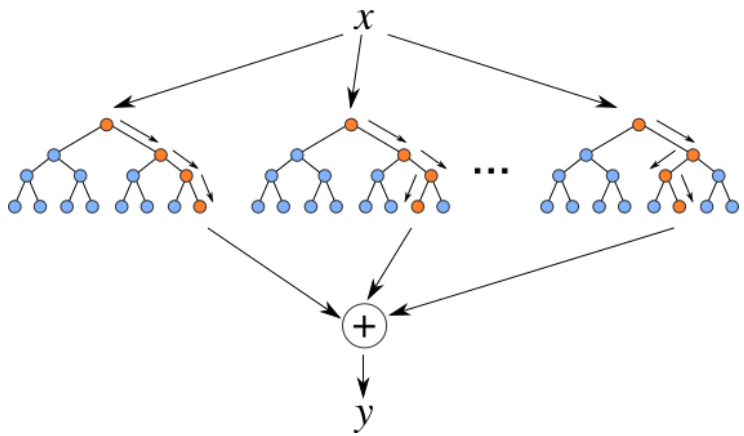
- *Regresión Lineal*
- *Arboles de decisión*
- *Support vector - Machine*



Algoritmos No supervisados

- *Clustering k-Means*





Por una universidad de excelencia y solidaria

Universidad del Cauca

CLR (CLUSTER-WISE LINEAR REGRESSION)

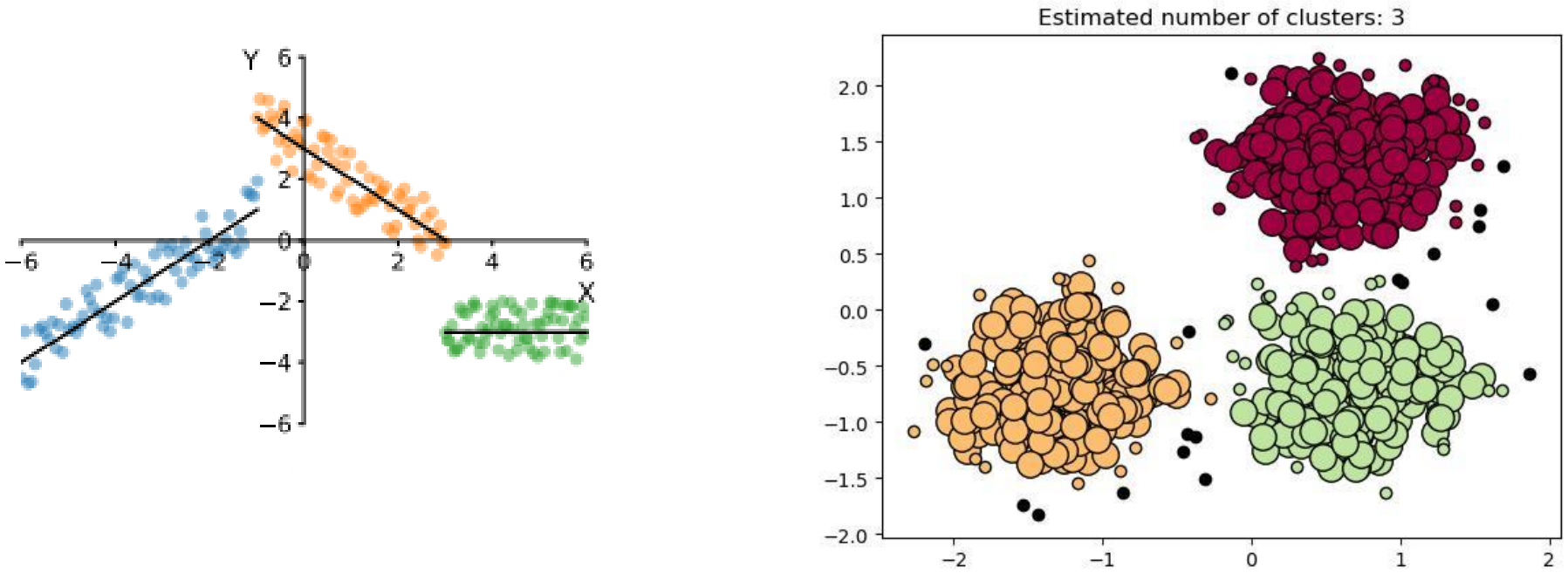
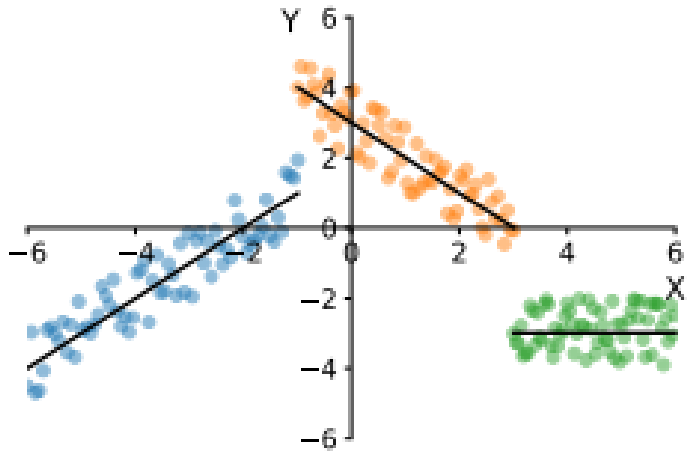


Fig 1: Algoritmo Clusterwise Linear Regression, tomado [1]

EVALUACÓN DE LA CALIDAD DEL ALGORTIMO CLR



Clsuter 1 → R-AJUSTADO 1

Cluster 2 → R-AJUSTADO 2

Cluster 3 → R-AJUSTADO 3

Coeficiente de determinación **[R^2]** determina que tan bien se ajustan los datos al modelo .[0,1]

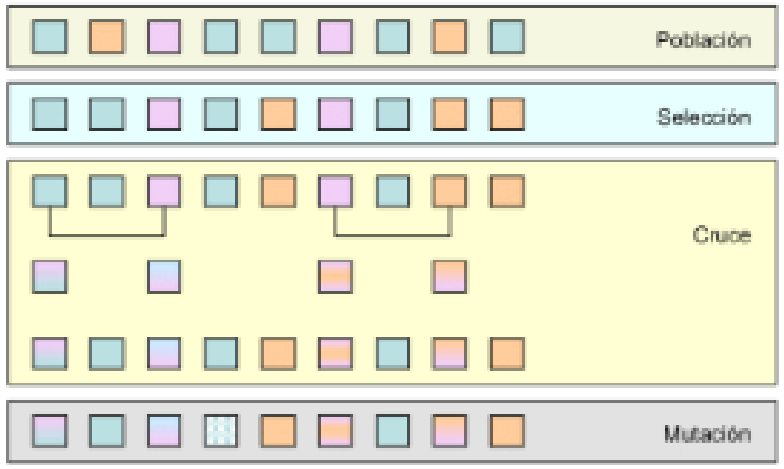
$$R - ADJ_{Total} = \frac{(R - ADJ_1 * NOCL1) + (R - ADJ_2 * NOCL2) + (R - ADJ_3 * NOCL1) + (R - ADJ_n * NOCLn)}{\#\ Total\ Observaciones}$$

Por una universidad de excelencia y solidaria

Universidad del Cauca

2. METAHURISTICA

ALGORTIMOS GENETICOS




```
2:  P ← {}
3:  for popsize times do
4:      P ← P ∪ {new random individual}
5:  Best ← □
6:  repeat
7:      for each individual P_i ∈ P do
8:          AssessFitness(P_i)
9:          if Best = □ or Fitness(P_i) > Fitness(Best) then
10:             Best ← P_i
11:     Q ← {}                                          ▷ Here's
12:     for popsize/2 times do
13:         Parent P_a ← SelectWithReplacement(P)
14:         Parent P_b ← SelectWithReplacement(P)
15:         Children C_a, C_b ← Crossover(Copy(P_a), Copy(P_b))
16:         Q ← Q ∪ {Mutate(C_a), Mutate(C_b)}
17:     P ← Q
18: until Best is the ideal solution or we have run out of time
19: return Best
```

Fig 2: pseudocodigo algortimo evolutivo, tomado de [5]

REPRESENTACIÓN DE LA SOLUCIÓN

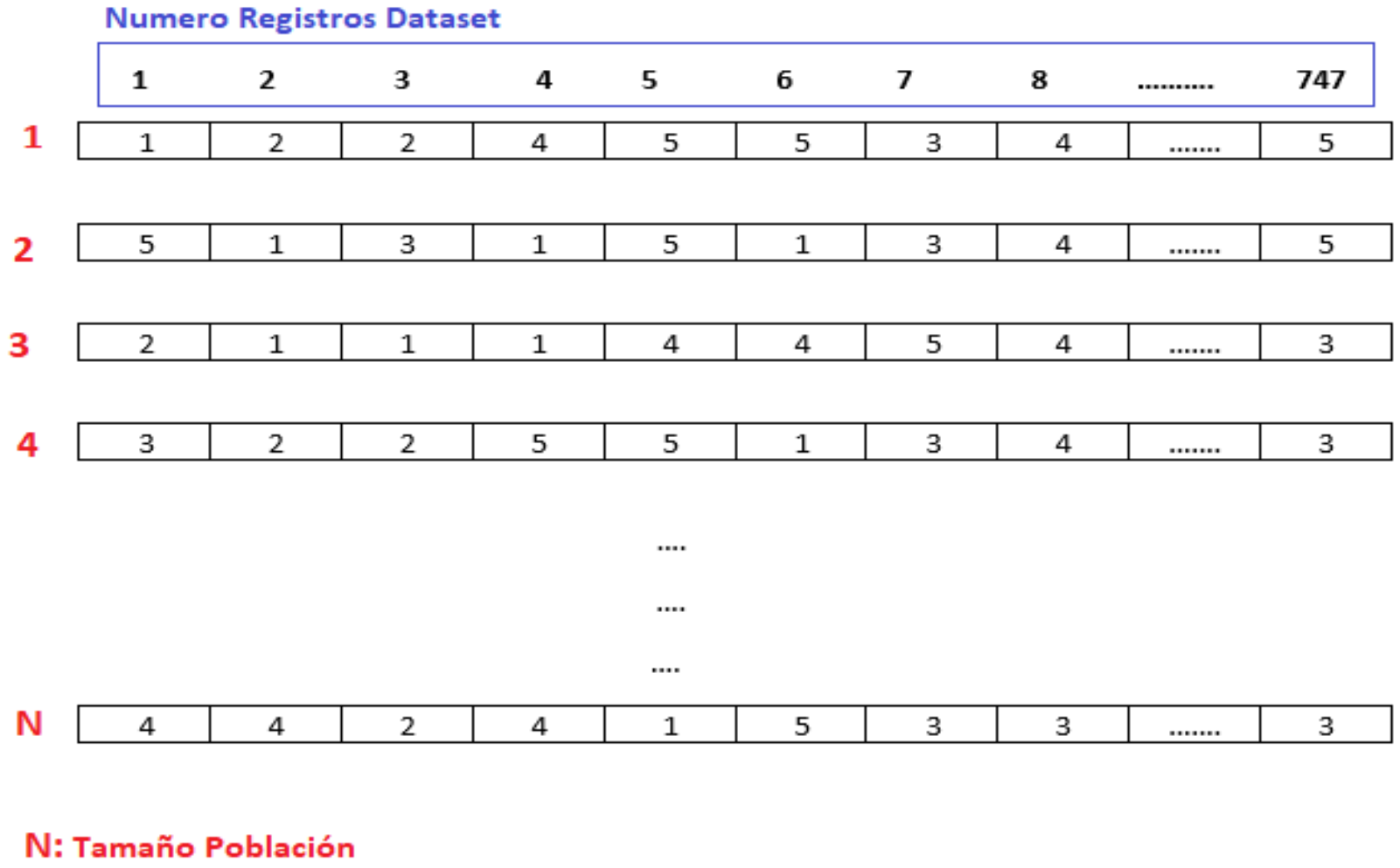
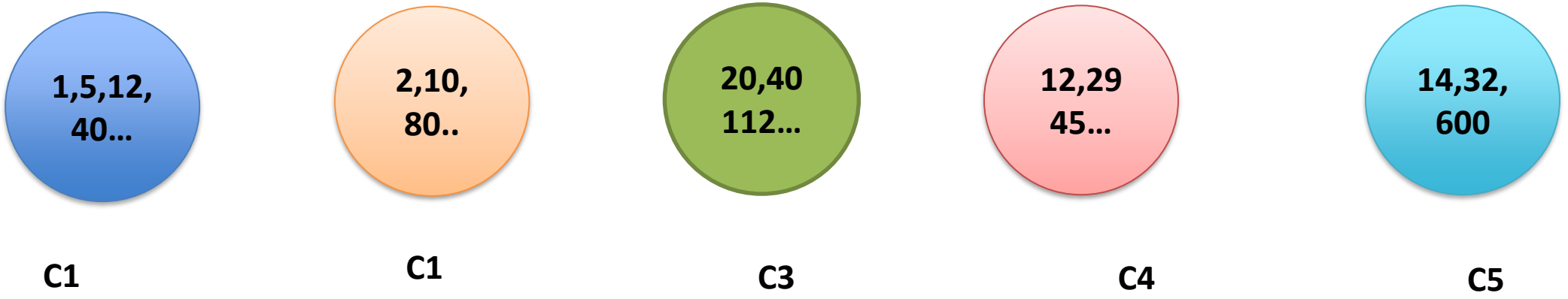


Fig 3: Solución discreta. Fuente propia

FUNCIÓN OBJETIVO

1	2	3	4	5	6	7	8	.....	747
1	2	2	4	5	5	3	4	.....	5



$$R - ADJ_{Total} = \frac{(R - ADJ_1 * NOCL1) + (R - ADJ_2 * NOCL2) + (R - ADJ_3 * NOCL1) + (R - ADJ_n * NOCLn)}{\#\ Total\ Observaciones}$$

FUNCIÓN OBJETIVO

```python
def fitnesIndividuo(df,individuo,Clusters,NEFO):

    listFitnes = []
    # Se realizan los respectivos filtros
    for i in range(Clusters):
        vectorArray = np.array(individuo)
        clus =  np.where(vectorArray == (i+1))
        list_cls = clus[0].flatten().tolist()
        # Se filtra el dataset deacuerdo a esos indices
        df_cls = df[df.index.isin(list_cls)==True]
        # Se realiza el llamado a la función lineal
        fitnes_cls = RegresionLineal(df_cls)
        listFitnes.append(fitnes_cls)

    # Fitnes solucion | Individuo
    FitnesSoluccion = sum(listFitnes)/len(df)
    # Actualizo el contador
    NEFO = NEFO+1
    return FitnesSoluccion,NEFO
```

Fig 4: Función Objetivo, Fuente Propia

Pasos

- Filtro el individuo para c/d cluster
- Obtengo el modelo de regresión lineal para los datos del cluster
- Obtengo el fitness de la solución, asociada c/d individuo.

FUNCIÓN OBJETIVO

```python
def RegresionLineal(df_cls):
    Pond_Radjustado = len(df_cls)
    # Se divide el dataset en conjunto de entrenamiento y pruebas [0.7 : Entrenamiento; 0.3: Testeo]
    data_train, data_test = train_test_split(df_cls, test_size=0.3, random_state=42)
    variables_independientes = '''~ C(MATERIAL_GENETICO) + C(TIPO_MATERIAL) +C(POSICION_PERFIL_RASTA) +C(ESTRUCTURA_RASTA)
                                    + C(OBSERVA_COSTRAS_BLANCAS_RASTA) + C(OBSERVA_COSTAS_NEGRAS_RASTA) + C(OBSERVA_PLANTAS
                                    + C(TIPO_SIEMBRA)+ C(SEM_TRATADAS) + C(DIAS_EN_FLORECER_A_COSECHAR) +ContEnfQui_Emer_Fl
                                    + ContPlaQui_Siem_Emer + PROFUND_RAICES_VIVAS_RASTA + Porc_Ar + Porc_FAr + Porc_BLANDO
    '''
    # Se crea el modelo
    mod_cls = smf.ols(formula='RDT_AJUSTADO'+variables_independientes, data=data_train).fit()
    #Me retorna el valor de la metrica de desempeño  para ese cluster.
    fitnes_cls = mod_cls.rsquared_adj * Pond_Radjustado
    return fitnes_cls
```

Fig 5: Modelo regresión Lineal, fuente propia

Pasos

- Divido los datos de c/d cluster en [Entrenamiento y Testeo]
- Obtengo el valor rsquared_adj

PSEUDOCODIGO

VARIABLES -PARAMETROS

```
popSize=100          # Se define el tamaño de la población
Clusters = 5         # Numero de Clusters
P = []               # Lista donde se va almacenar la poblacion.
FitnesPobl = []      # Lista vacia que almacena Valores FO
Best = []            # Mejor solucion de ajuste
QBest = 0            # Calidad del best Inicial
NEFO = 0             # Contador Numero de Evaluaciones FO
nDim = len(df)       #Numero de Observaciones del dataset.

# Crea la poblacion
P,FitnesPobl,NEFO = crearPoblacion(popSize, nDim, Clusters,NEFO)
```

```
repeat
    for each individuo (pi) C P do:
         if (len(Best) == 0) or (FitnesPobl[z] > QBest):
            Best = P[z]
            QBest = FitnesPobl[z]

    if (NEFO == 5000):
        break

    Q = []
    FitnesQ = []
    for i in range(int(popSize/2)):
        # Seleccion padres
        Padre1,Padre2 = seleccionPadres(popSize,P)
        # Cruce
        c1,c2 = cruzeHijos(Padre1,Padre2)
        # Mutacion
        Mc1,Mc2,QMc1,QMc2,NEFO = mutacionIntercambio(c1,c2,NEFO)
        # Agregamos los cruzes a la nueva Poblacion
        Q.append(Mc1)
        Q.append(Mc2)
        # Se obtienen los Fitnes de la nueva Poblacion
        FitnesQ.append(QMc1)
        FitnesQ.append(QMc2)

    # Remplazo la nueva poblacion y la Funcion de ajuste
    P = Q
    FitnesPobl  = FitnesQ

Hasta Best solucion ideal (Best < 0.9) or NEFO = 5000
```

ALGUNAS FUNCIONES

```python
# Funcion para crear la poblacion
# ================================================================================
def crearPoblacion(popSize,nDim,Clusters,NEFO):
    for i in range(popSize):
        #print(i)
        # Creo el individuo
        individuo = vectorSolution(nDim, Clusters)
        P.append(individuo)
        # Evaluo el individuo (Fitness)
        FitInd = fitnesIndividuo(df,individuo,Clusters,NEFO)
        # Agrego el ajuste a lista de Fitness de la población
        NEFO = FitInd[1]
        FitnesPobl.append(FitInd[0])


    return P,FitnesPobl,NEFO
```

Fig 6: función crear población, fuente propia

ALGUNAS FUNCIONES

```python
# Funcion  Seleccion de Padres  (Aleatoria)     # paso poblacion
# ============================================================================
def seleccionPadres(popSize,Poblacion):
    # Se genrar una lista con el Tamaño de la poblacion
    tamanoPoblacion = list(range(popSize))
    padres = np.random.choice(tamanoPoblacion, 2, False)
    Padre1 = Poblacion[padres[0]]
    Padre2 = Poblacion[padres[1]]

    return Padre1,Padre2
```

Fig 7. Función selección padres, fuente propia

ALGUNAS FUNCIONES

```python
# Funcion para remplazar la nueva Poblacion  - Operador del peor
# ================================================================================
def RemplazoDelPeor(fitP,fitNP,Poblacion,NuevaPoblacion):
    # Agrupanos en una sola lista el fitness de ambas poblaciones
    fitT = fitP + fitNP
    # Agrupamos en una sola lista las poblaciones
    PT = Poblacion + NuevaPoblacion
    # Se convierte la lista de Fitnes Total  a df
    df =  pd.DataFrame(fitT,columns=["Fitnes"])
    # Obtengo los N mejores individuos de acuerdo al tamaño de la población
    df = df.sort_values('Fitnes',ascending=False)[0:len(fitP)]
    # Lista de los mejores valores de Aptitud
    Nfit = list(df.Fitnes)
    # Lista de los indices de la Poblacioón
    indexNuevaPoblacion = list(df.index)
    # Ubicamos los Individuos de la Nueva Población

    NP = []
    for i in range(len(indexNuevaPoblacion)):
        NP.append(PT[indexNuevaPoblacion[i]])

    # Retorna la lista de la nueva Poblacion y los respectivos Fitnes
    return NP, Nfit
```

Fig 8. Función remplazo del peor, fuente propia

4. RESULTADOS

CONFIGURACION DE ALGORTIMOS

Algoritmos	Operadores			
	Selección	**Cruce**	**Mutación**	**Remplazo**
A1	Aleatoria	Un Punto	Intercambio	De Padres
A2	ELitismo	Un punto	Intercambio	Del peor

Tabla 1: Tabla de Configuraciones de algoritmos

ANALISIS DE RESULTADOS

Algoritmo 1	
Población	**Fitness**
50	0.6249
100	0.6379
200	0.6384

Tabla 2: Resultados configuración 1

Algoritmo 2	
Población	**Fitness**
50	0.6047
100	0.6284
200	**0.6512**

Tabla 3: Resultados configuración 1

5.CONCLUSIONES

- Para el calculo de la función objetivo (fitness) y crear cada uno de los submodelos (Regresión Lineal) únicamente se tuvo en cuenta las 30 características mas significativas del dataset. Seria importante analizar el desempeño del modelo incluyendo mas características (Categóricas y numéricas).

- La solución **Qbest** nos muestra el mejor valor de ajústate para el macromodelo, teniendo en cuenta los valores de ajuste (R^2) de cada uno de los micromodelos.

- La implementación de los algoritmos genéticos se realiza con el fin de determinar la mejor distribución de las observaciones en cada uno de los clusters, de tal manera que se maximice la función objetivo.

6.BIBLIOGRAFIA

[1]https://www.researchgate.net/publication/324859471_Novel_Prediction_Techniques_Based_on_Clusterwise_Linear_Regression
[2]Algoritmos genéticos teoría, https://conogasi.org/articulos/algoritmos-geneticos/
[3]Librerías Python, https://www.programiz.com/python-programming/methods/list/index
[4] http://www.cs.us.es/~fsancho/?e=65
[5] Diapositivas de Clase , Materia Metahuristicas.

Por una universidad de **excelencia** y **solidaria**

Universidad del Cauca

¡Gracias por su atención!

www.unicauca.edu.co