Universidad Tecnológica Nacional

Facultad Regional Córdoba

Ingeniería en Sistemas de información

Ciencia de Datos

Ignite Your Mind (IYM)

“Segunda Entrega”

Grupo 3

Integrantes:

- **82217 Anzulovich Garzón, Valentina - valenanzu.utn@gmail.com**
- **80359 Berrotarán, María Luz - luzberrotaran@gmail.com**
- **78678 Salas, Bruno Matías - brunosalas1@hotmail.com**
- **82508 Sonzini Astudillo, Enrique José - enriquesonzini@gmail.com**
- **81841 Yarbouh, Yamili - yamiliyarbouh@gmail.com**

Docentes:

- **Pablo Alberto Sacco**
- **Franco Mana**
- **Marisa del Carmen Callejas**

Curso: 5K2

Fecha de presentación: 16/10/2024

Índice

Introducción

En esta segunda entrega trabajamos los datos para ajustar los valores de las diferentes tablas y de este modo tener un set de datos realmente unificado. Comenzamos por definir criterios para la adaptación de los datos a un formato común, para esto fueron necesarias varias transformaciones que vamos a estar detallando. Todo esto fue fundamental para lograr nuestro objetivo que fue la realización de el ETL.

Desarrollo

Como ya hemos mencionado en la entrega anterior, vamos a estar utilizando el dataset de las pruebas “Aprender” correspondientes a los años 2019 y 2022 con los datos obtenidos de la página del gobierno.

Empezamos trabajando cada año por separado, que es cómo los datos estaban originalmente, para evitar inconsistencias. Por lo tanto para esta primera parte vamos a ejemplificar con el año 2019.

Aclaración: La indentación en algunos casos fue modificada al momento de tomar las capturas para lograr mayor legibilidad.

```
#Invocación del archivo csv
url_2019 = 'https://drive.google.com/uc?export=download&id=10iOfvQy79727IttWiOSxOXkcFTc9NEY2'
output_2019 = 'data_2019.csv'
gdown.download(url_2019, output_2019, quiet=False)

#Carga del archivo csv al dataframe df_2019
df_2019 = pd.read_csv(output_2019,sep=';')
```

Eliminamos las columnas que encontramos que no eran comunes a los años en cuestión

```
columns_to_remove = ["tiene_smart_TV", "tiene_microondas", "tiene_aire_acondicionado", "familia_estudiante_universitario", \
                     "familia_graduado_universitario", "cuida_hermano", "acuerdo_buena_convivencia", "acuerdo_llevarse_bien_estudiantes", \
                     "acuerdo_llevarse_bien_docentes", "acuerdo_sentirse_bien", "frecuencia_peleas", "frecuencia_ataque_a_profesores", \
                     "frecuencia_ataque_a_alumnos_rrss", "frecuencia_ataque_a_profesores_rrss", "frecuencia_daños", \
                     "acuerdo_convivencia_respetado_alumnos", "acuerdo_convivencia_respetado_docentes", "acuerdo_convivencia_se_tomas_medidas", \
                     "acuerdo_convivencia_aula", "conflicto_tratado_otro", "escuela_orientacion_vocacionlan_carrera","escuela_orientacion_vocacionlan_carrera_asistio",\
                     "escuela_informacion_carreras_1","escuela_informacion_carreras_1_asistio",\
                     "escuela_informacion_carreras_2","escuela_informacion_carreras_2_asistio","escuela_informacion_carreras_3",\
                     "escuela_informacion_carreras_3_asistio","pregunta_irrelevante_1","pregunta_irrelevante_2","pregunta_irrelevante_3",\
                     "pregunta_irrelevante_4","pregunta_irrelevante_5","pregunta_irrelevante_6","pregunta_irrelevante_7","pregunta_irrelevante_8",\
                     "pregunta_irrelevante_9","pregunta_irrelevante_10","pregunta_irrelevante_11","alumno_hacer_despues_secundario",\
                     "pregunta_irrelevante_12","pregunta_irrelevante_13","pregunta_irrelevante_14", "pregunta_irrelevante_15", \
                     "pregunta_irrelevante_16", "pregunta_irrelevante_17", "pregunta_irrelevante_18", "pregunta_irrelevante_19",\
                     "pregunta_irrelevante_20", "pregunta_irrelevante_21", "pregunta_irrelevante_22", "pregunta_irrelevante_23",\
                     "pregunta_irrelevante_24", "pregunta_irrelevante_25", "pregunta_irrelevante_26", "pregunta_irrelevante_27", \
                     "pregunta_irrelevante_28", "pregunta_irrelevante_29", "pregunta_irrelevante_30", "pregunta_irrelevante_31",\
                     "pregunta_irrelevante_32", "pregunta_irrelevante_33", "pregunta_irrelevante_34", "pregunta_irrelevante_35",\
                     "pregunta_irrelevante_36", "pregunta_irrelevante_37", "pregunta_irrelevante_38", "pregunta_irrelevante_39", \
                     "pregunta_irrelevante_40", "pregunta_irrelevante_41", "pregunta_irrelevante_42", "pregunta_irrelevante_43", \
                     "pregunta_irrelevante_44", "pregunta_irrelevante_45", "pregunta_irrelevante_46", "pregunta_irrelevante_47", \
                     "pregunta_irrelevante_48", "pregunta_irrelevante_49", "pregunta_irrelevante_50", "habilidad_comprender_texto", \
                     "habilidad_escribir_texto", "habilidad_expocicion_oral", "habilidad_resolver_ejercicios", "acuerdo_disfruta_matematica", \
                     "acuerdo_interesa_matematica", "acuerdo_esfuerzo_matematica", "acuerdo_proponer_bien_matematica", \
                     "acuerdo_matematica_importante", "acuerdo_aprendi_mucho_matematica", "acuerdo_investigar_redactar_informe", \
                     "acuerdo_tomar_apuntos", "acuerdo_porcision_propia", "acuerdo_resolver_situaciones", "acuerdo_presentaciones_orales", \
                     "acuerdo_identificar_dificultad_aprendizaje", "acuerdo_expresar_ideas", "acuerdo_escuchar_demas", \
                     "acuerdo_analizar_consecuencias_acciones", "acuerdo_asumir_compromiso", "acuerdo_trabajo_grupo", \
                     "escuela_proporciona_clases_apoyo", "escuela_seguimiento_personalizado", "escuela_adapta_clases", \
                     "escuela_diferentes_formas_evaluar", "enseñar_sobre_desarrollo_sustentable", "enseñar_sobre_oferta_educativa", \
                     "enseñar_sobre_otros", "frecuencia_uso_computadora", "frecuencia_uso_notebook", "frecuencia_uso_tablet", \
                     "frecuencia_uso_carro_digital", "frecuencia_uso_celular", "usa_celular_aula", "uso_internet_como_buscar", \
                     "uso_internet_como_cuidar_datos", "uso_internet_como_funciona_busqueda", "uso_internet_como_citar", \
                     "uso_internet_identificar_proposito", "uso_compu_clase_escribir", "uso_compu_clase_leer", "uso_compu_clase_cuestionarios", \
                     "uso_compu_clase_buscar_informacion", "uso_compu_clase_simulaciones", "uso_compu_clase_animaciones", \
                     "uso_compu_clase_multimedia", "uso_compu_clase_juegos", "uso_compu_clase_programar_1", "uso_compu_clase_robots", \
                     "uso_compu_clase_excel", "uso_compu_clase_programar_2", "uso_compu_materia_lengua", "uso_compu_materia_historia", \
                     "uso_compu_materia_matematica", "uso_compu_materia_plastica", "uso_compu_materia_quimica_fisica", "uso_compu_materia_biologia", \
                     "uso_compu_materia_informatica", "uso_compu_materia_otros", "uso_compu_materia_ninguna", "edad_uso_compu_1", "edad_uso_compu_2", \
                     "dispositivo_buscar_informacion", "dispositivo_estudiar", "dispositivo_tareas_grupo", "dispositivo_tareas_multimedia", \
                     "dispositivo_videos_temas_escuela", "frecuencia_chatear", "frecuencia_buscar_informacion_internet", "frecuencia_hacer_multimedia",\
                     "frecuencia_ver_pelicula", "frecuencia_rrss", "frecuencia_leer_mails", "frecuencia_participar_foro", "frecuencia_aprender_idioma",\
                     "frecuencia_leer_noticias", "frecuencia_jugar", "ponder", "lpondera", "mpondera", "ldesemp", "mdesemp",\
                     "modelo", "isocioa_puntaje", "isocioa", "isocioal_puntaje", "isocioal", "isocioam_puntaje", "isocioam", "repitencia_dicotomica",\
                     "jardín", "ap37_dicotomica", "ap29_01.dico", "ap29_02.dico", "ap29_03.dico", "ap29_04.dico", "ap29_05.dico", "ap29_06.dico",\
                     "ap29_07.dico", "ap26_rec", "trabaja_fuera_hogar", "trabaja_fuera_hogar_remunerado", "migración", "edadA_junio2019", "sobreedad",\
                     "infraestructura", "iinfraestructura", "ap42_01rec", "ap42_02rec", "ap42_03rec", "ap42_04rec", "escuela_proporciona_clases_"]

df_2019 = df_2019.drop(columns=columns_to_remove, errors='ignore')
```

Columnas comunes

Las columnas que se mantuvieron fueron las siguientes 70:

ID1
cod_provincia
sector
ambito
seccion
idalumno
sexo
nacionalidad_alumno
vive_madre
vive_padre
vive_hermano
vive_hijo
vive_tio
vive_abuelo
vive_pareja
vive_amigos
vive_otro
tiene_hijos

habitaciones_en_casa
tiene_auto
tiene_baño
tiene_computadora
tiene_tablet
tiene_internet
celular_propio
celular_propio_internet
libros_casa
nivel_educativo_madre
nivel_educativo_padre
cuida_familiar
realiza_tareas_hogar
ayuda_trabajo_padres
dias_trabajo_fuera_casa
trabaja_remunerado
fuera_escuela_junto_amigos
fuera_escuela_idioma
fuera_escuela_leer
fuera_escuela_deporte
fuera_escuela_clases_artes
fuera_escuela_television
asistio_jardin
repitio_primaria
repitio_secundaria_1
repitio_secundaria_2
faltas
motivo_faltas_enfermedad
motivo_faltas_ganas
motivo_faltas_ayuda_casa
motivo_faltas_problemas_clima
motivo_faltas_trabajo
motivo_faltas_otros
hay_acuerdo_convivencia
alumno_conoce_acuerdo_convivencia
acuerdo_convivencia_participacion_alumnos_definicion
conflicto_trado_adultos
conflicto_tratado_involucrados
conflicto_tratado_equipo_orientacion
conflicto_tratado_dejarlo_pasar

conflicto_tratado_sanciones
enseñar_sobre_uso_tecnologias
enseñar_sobre_cambio_climatico
enseñar_sobre_salud
enseñar_sobre_conflictos_internacionales
enseñar_sobre_demanda_laboral
lpuntaje
mpuntaje
padres_extranjeros
personas_en_casa
bullying_aspecto_fisico
edad

Una vez que las tuvimos seleccionadas, las transformamos a formato numérico para poder comenzar a trabajar con ellas.

```
for column in df_2019.columns:
  try:
    #df_2019[column] = pd.to_numeric(df_2019[column], errors='coerce')
    df_2019[column] = pd.to_numeric(df_2019[column].str.replace(',', '.'), errors='coerce')
  except:
    pass
```

Luego comenzamos a adaptar los datos de las columnas restantes para que fueran consistentes en ambas tablas

Columnas con datos adaptados:

Nacionalidad del alumno:
Se adaptó la de 2019 a los valores de 2022

```
df_2019['nacionalidad_alumno'] = df_2019['nacionalidad_alumno']
.replace([5, 6, 7, 8, 9, 10, 15], [7, 8, 9, 5, 10, 14, 14])
```

Padres extranjeros:
En 2019 fue necesario crear esta columna basándonos en los datos de padre extranjero o madre extranjera, dando un valor de uno cuando alguno de los padres lo fuera y cero en caso contrario.

```python
def fusionar_nacionalidades(row):
  nacionalidad_madre = row['nacionalidad_madre']
  nacionalidad_padre = row['nacionalidad_padre']

  if nacionalidad_madre < 0 and nacionalidad_padre < 0:
    return -9
  elif nacionalidad_madre == 1 or nacionalidad_padre == 1:
    return 1
  else:
    return 0

df_2019['padres_extranjeros'] = df_2019.apply(fusionar_nacionalidades, axis=1)
df_2019 = df_2019.drop(['nacionalidad_madre', 'nacionalidad_padre'], axis=1)
```

Personas en casa: Sumamos en esta pregunta el hecho de si vive solo, que se encontraba en una columna aparte.

```python
# Crear una nueva columna 'personas_en_hogar' con valores 0 inicialmente
df_2019['personas_en_hogar'] = 0

# Asignar 1 a la columna 'personas_en_hogar' si vive_solo es 1
df_2019.loc[df_2019['vive_solo'] == 1, 'personas_en_hogar'] = 1

# Asignar el valor de (personas_en_casa + 1) a la columna 'personas_en_hogar' si vive_solo es 0
df_2019.loc[df_2019['vive_solo'] != 1, 'personas_en_hogar'] = df_2019['personas_en_casa'] + 1

print(df_2019[['personas_en_casa','vive_solo', 'personas_en_casa']].head(10))

# Eliminar las columnas originales 'vive_solo' y 'personas_en_casa' (opcional)
df_2019 = df_2019.drop(['vive_solo', 'personas_en_casa'], axis=1)

#Renombrar la columna
df_2019 = df_2019.rename(columns={'personas_en_hogar': 'personas_en_casa'})

# Mostrar las primeras filas del DataFrame para verificar la fusión
print(df_2019[['personas_en_casa']].head(10))
```

Vive con su madre, Vive con su padre, vive con su hermano, vive con su hijo, vive con su tío, vive con su abuelo, vive con su pareja, vive con amigos, vive con otro:
Las preguntas referidas a con quién vive que eran 2 opciones, las pasamos a codificación binaria para unificar los criterios.

```python
columns_to_replace = ['vive_madre', 'vive_padre', 'vive_hermano', 'vive_hijo',
                      'vive_tio', 'vive_abuelo', 'vive_pareja', 'vive_amigos', 'vive_otro']

for column in columns_to_replace:
  df_2019[column] = df_2019[column].replace(-9, 0)
```

Tiene hijos:
También fueron transformadas a codificación binaria.

```
df_2019['tiene_hijos'] = df_2019['tiene_hijos'].replace(2, 0)
```

Tiene auto, tiene baño, tiene computadora, tiene tablet, tiene internet:
Las pasamos a codificación binaria para unificar criterio.

```
columns_to_transform = ['tiene_auto', 'tiene_baño', 'tiene_computadora', 'tiene_tablet']

for column in columns_to_transform:
  df_2019[column] = df_2019[column].apply(lambda x: x - 1 if x > 0 else x)
  df_2019[column] = df_2019[column].apply(lambda x: 1 if x > 0 else x)
```

```
df_2019['tiene_internet'] = df_2019['tiene_internet'].replace(2, 0)
```

Tiene celular propio, tiene internet en su celular:
Las pasamos a codificación binaria para unificar criterio y pasamos a -9 los valores inválidos.

```
df_2019['celular_propio'] = df_2019['celular_propio'].replace(2, 0)
df_2019['celular_propio_internet'] = df_2019['celular_propio_internet'].replace(2, 0)
df_2019.loc[df_2019['celular_propio_internet'] > 2, 'celular_propio_internet'] = -9
```

Nivel educativo padre, nivel educativo madre:
Creamos un diccionario de mapeo para la transformación de valores, luego cambiamos los 4 por 0.

```
mapping = {2: 1, 3: 1, 4: 2, 5: 3, 6: 3, 7: 3, 1: 4, 8: 4}

# Aplicar el mapeo a las columnas 'nivel_educativo_madre' y 'nivel_educativo_padre'
df_2019['nivel_educativo_madre'] = df_2019['nivel_educativo_madre'].map(mapping)
df_2019['nivel_educativo_padre'] = df_2019['nivel_educativo_padre'].map(mapping)
```

```
df_2019['nivel_educativo_padre'] = df_2019['nivel_educativo_padre'].replace(4, 0)
df_2019['nivel_educativo_madre'] = df_2019['nivel_educativo_madre'].replace(4, 0)
```

Cuida familiar, realiza tareas del hogar: Creamos un diccionario de mapeo para la transformación de valores y que coincidan con los datos de 2022.

```
mapping = {4: 1, 1: 2, 2: 2, 3: 2}

# Apply the mapping to the 'cuida_familiar' and 'realiza_tareas_hogar' columns
df_2019['cuida_familiar'] = df_2019['cuida_familiar'].map(mapping)
df_2019['realiza_tareas_hogar'] = df_2019['realiza_tareas_hogar'].map(mapping)
```

Vivió situaciones de bullying: Creamos la columna de bullying por aspecto físico con los datos de las columnas relacionadas con el bullying.

```
df_2019['bullying_aspecto_fisico'] = 0
```

En el caso de haber sufrido algún tipo de bullying por su aspecto físico la marcamos con 1

```
df_2019.loc[(df_2019['frecuencia_bullying_aspecto_fisico_1'] == 1)
 | (df_2019['frecuencia_bullying_aspecto_fisico_2'] == 1),
            'bullying_aspecto_fisico'] = 1
```

Eliminamos las columnas originales

```
df_2019 = df_2019.drop(['frecuencia_bullying_aspecto_fisico_1',
                        'frecuencia_bullying_aspecto_fisico_2'], axis=1)
```

Edad: Con el año y mes de nacimiento calculamos la edad.
Primero obtuvimos el año a partir de los valores marcados en las opciones.

```
# prompt: En la columna año_nacimiento, cambiar 1 por 1998, 2 por 1999,
# 3 por 2000, 4 por 2001, 5 por 2002, 6 por 2003 y el resto por null

def convert_year(year):
  if year == 1:
    return 1998
  elif year == 2:
    return 1999
  elif year == 3:
    return 2000
  elif year == 4:
    return 2001
  elif year == 5:
    return 2002
  elif year == 6:
    return 2003
  else:
    return None

df_2019['año_nacimiento'] = df_2019['año_nacimiento'].apply(convert_year)
```

Una vez hecho esto, pudimos crear una función que calcule la edad.

```
# Crear una función para calcular la edad
def calcular_edad(row):
  try:
    fecha_nacimiento = datetime(int(row['año_nacimiento']), int(row['mes_nacimiento']), 1)
    fecha_referencia = datetime(2019, 9, 1)
    edad = (fecha_referencia - fecha_nacimiento).days // 365
    return edad
  except (ValueError, TypeError):
    return None

# Aplicar la función a cada fila del DataFrame para calcular la edad y crear una nueva columna
df_2019['edad'] = df_2019.apply(calcular_edad, axis=1)

print(df_2019[['mes_nacimiento', 'año_nacimiento', 'edad']].head())

# Eliminar las columnas 'mes_nacimiento' y 'año_nacimiento'
df_2019 = df_2019.drop(['mes_nacimiento', 'año_nacimiento'], axis=1)

# Mostrar las primeras filas del DataFrame con la nueva columna 'edad'
print(df_2019[['edad']].head())
```

Ayuda a sus padres en el trabajo, trabaja remunerado:
Lo pasamos a codificación binaria.

```
df_2019['ayuda_trabajo_padres'] = df_2019['ayuda_trabajo_padres'].replace(2, 0)
df_2019['trabaja_remunerado'] = df_2019['trabaja_remunerado'].replace(2, 0)
```

Trabaja fuera de casa, fuera de la escuela se junta con amigos, fuera de la escuela hace algún idioma, fuera de la escuela lee, fuera de la escuela hace deportes, fuera de la escuela va a clase de arte, fuera de la escuela ve televisión:

Transformamos todos estos valores a codificación binaria, pasando la pregunta de cuántos días trabaja a si trabaja o no, unificando criterios.

```
columns_to_replace = [ 'dias_trabajo_fuera_casa', 'fuera_escuela_junto_amigos',
                      'fuera_escuela_idioma', 'fuera_escuela_leer', 'fuera_escuela_deporte',
                       'fuera_escuela_clases_artes', 'fuera_escuela_television']

for column in columns_to_replace:
 df_2019[column] = df_2019[column].replace(1, 0)
 df_2019[column] = df_2019[column].replace(2, 1)
 df_2019[column] = df_2019[column].replace(3, 1)
 df_2019[column] = df_2019[column].replace(4, 1)
 df_2019[column] = df_2019[column].replace(5, 1)
 df_2019[column] = df_2019[column].replace(6, 1)
```

Asistió al jardín, repitió primaria, repitió secundaria:
Adaptamos la codificación binaria reemplazando el 4 por 0 si no asistió a jardín.

```
df_2019['asistio_jardin'] = df_2019['asistio_jardin'].replace(4, 0)
```

El resto de los valores los adaptamos a la codificación de 2022.

```
# prompt: en las columnas repitio_primaria, repitio_secundaria_1, repitio_secundaria_2
#restarles 1 a cada fila si el valor es menor a 3

columns_to_modify = ['repitio_primaria', 'repitio_secundaria_1', 'repitio_secundaria_2']

for column in columns_to_modify:
  df_2019.loc[df_2019[column] < 3, column] = df_2019.loc[df_2019[column] < 3, column] -1
```

Conflicto tratado con adultos, conflicto tratado entre involucrados, conflicto tratado con equipo de orientación, conflicto tratado dejándolo pasar, conflicto tratado con sanciones:
Ajustamos las notaciones para que coincidan con las de 2022.

```
columns_to_replace = ['conflicto_trado_adultos', 'conflicto_tratado_involucrados',
                      'conflicto_tratado_equipo_orientacion',
                      'conflicto_tratado_dejarlo_pasar', 'conflicto_tratado_sanciones']

for column in columns_to_replace:
  df_2019[column] = df_2019[column].replace(-9, 0)
```

Motivo de faltas enfermedad, motivo de faltas ganas, motivo de faltas por ayudar en casa, motivo de faltas clima, motivo de faltas trabajo, motivo de faltas otros:

Ajustamos las notaciones para que coincidan con las de 2022.

```
columns_to_replace = ['motivo_faltas_enfermedad', 'motivo_faltas_ganas',
                      'motivo_faltas_ayuda_casa', 'motivo_faltas_problemas_clima',
                      'motivo_faltas_trabajo', 'motivo_faltas_otros']

for column in columns_to_replace:
  df_2019[column] = df_2019[column].replace(-9, 0)
```

Si creen que es necesario enseñar sobre el uso de la tecnología, enseñar sobre el cambio climático, enseñar sobre salud, enseñar sobre conflictos internacionales, enseñar sobre demandas laborales:

Ajustamos las notaciones para que coincidan con las de 2022.

```
columns_to_replace = ['enseñar_sobre_uso_tecnologias',
                      'enseñar_sobre_cambio_climatico',
                      'enseñar_sobre_salud',
                      'enseñar_sobre_conflictos_internacionales',
                      'enseñar_sobre_demanda_laboral']

for column in columns_to_replace:
  df_2019[column] = df_2019[column].replace(-9, 2)
```

Hay acuerdo de convivencia, el alumno conoce el acuerdo de convivencia, los alumnos participan en su definición:

Cambiamos la codificación de las columnas alumno_conoce_acuerdo_convivencia y acuerdo_convivencia_participacion_alumnos_definicion

```
columns_to_replace = ['alumno_conoce_acuerdo_convivencia',
                      'acuerdo_convivencia_participacion_alumnos_definicion']

for column in columns_to_replace:
  df_2019.loc[df_2019[column] == 1, column] = 2
  df_2019.loc[(df_2019[column] == 3) | (df_2019[column] == 4), column] = 1
```

Pasamos a codificación binaria, cambiando los 2 por 0

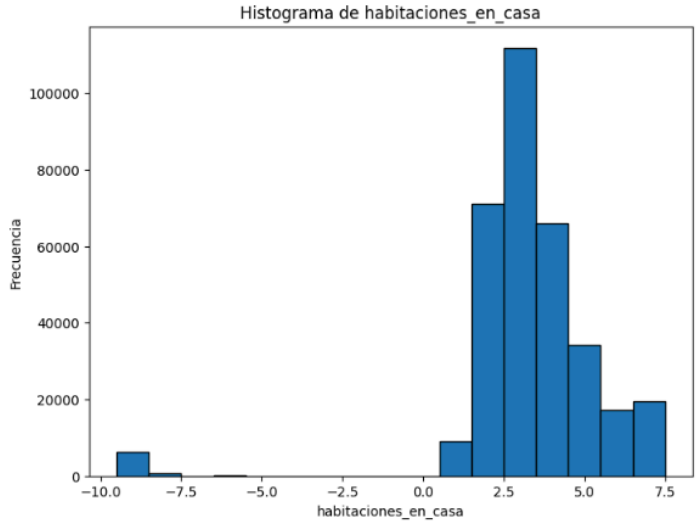
```
columns_to_replace = ['hay_acuerdo_convivencia', 'alumno_conoce_acuerdo_convivencia',
                      'acuerdo_convivencia_participacion_alumnos_definicion']

for column in columns_to_replace:
  df_2019[column] = df_2019[column].replace(2, 0)
```

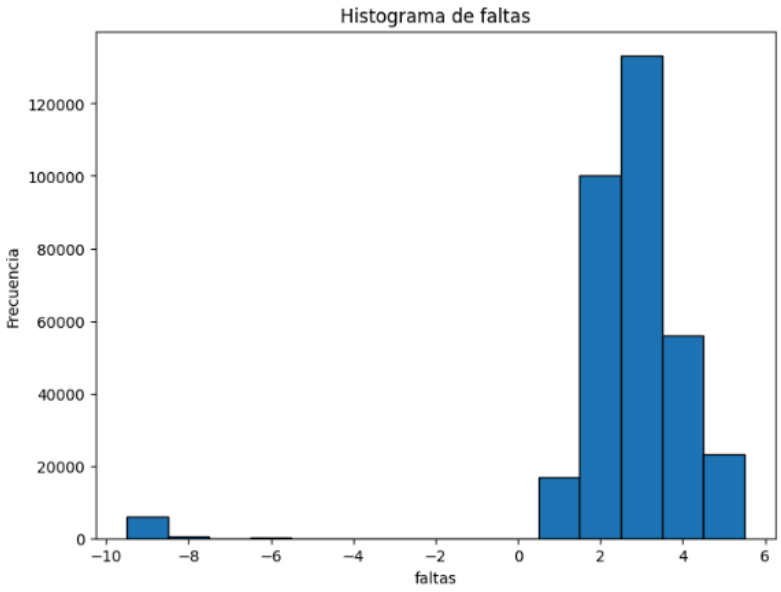
Comprobaciones intermedias

A medida que se fueron realizando los cálculos antes mencionados, íbamos corroborando que los resultados fueran más o menos los esperados.

Representación de los datos con Negativos

Con los histogramas de frecuencia pudimos tener una idea de la tendencia de los datos de cada columna. Los números negativos representan los datos inválidos, ya sea porque no marcan o marcan más de una opción. Estos son algunos ejemplos:





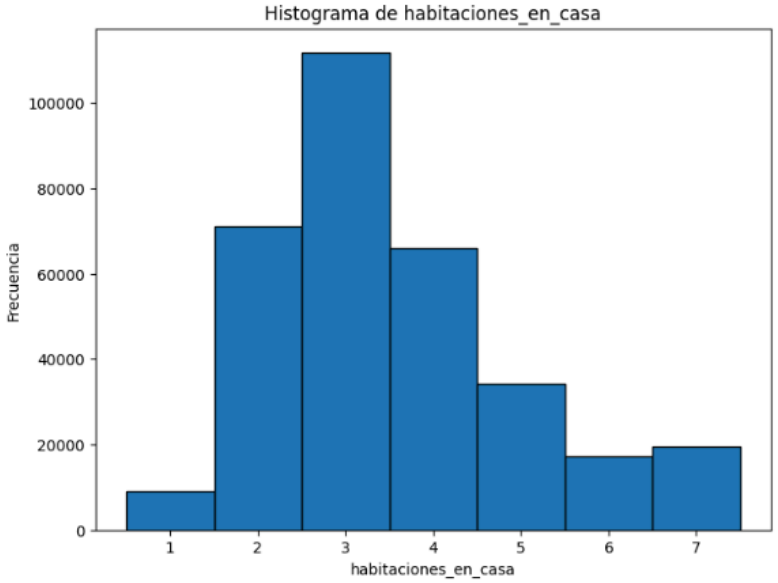
Transformar negativos en Null

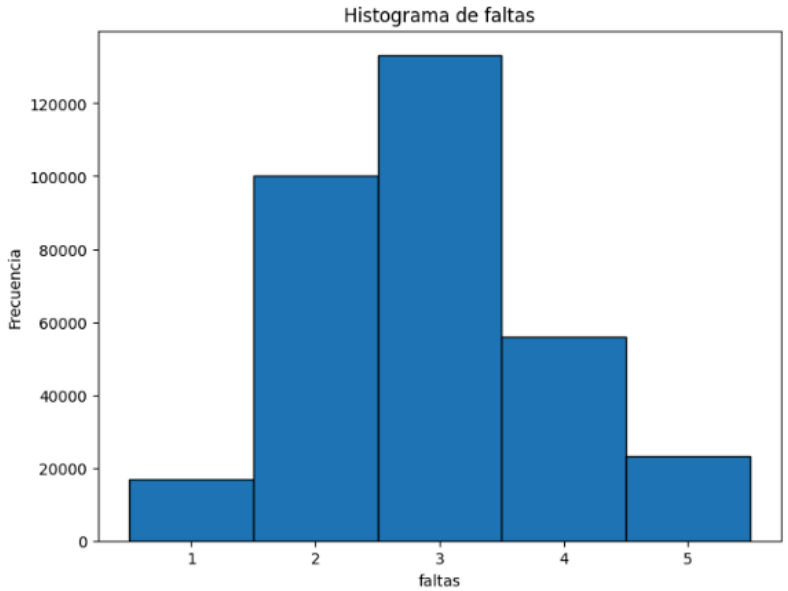
El siguiente paso fue transformar todos los valores negativos a Nulls, para su posterior eliminación.

```
for column in df_2019.columns:
  try:
    df_2019.loc[df_2019[column] <= -1, column] = np.nan
  except:
    pass
```

Representación con Nulls

Una vez que dejamos de lado aquellos datos que no aportan valor, como los null que vimos en la gráfica, se distribuyó diferente en los ejes. Se puede ver si analizamos las mismas columnas que antes.





Significatividad de los nulos en 2019

Teníamos que averiguar en qué medida los nulos afectaban nuestros resultados para ver qué es lo que podríamos llegar a hacer en cada caso.

```python
# Contar la cantidad de valores nulos en cada columna
null_counts = df_2019.isnull().sum()

# Mostrar la cantidad de valores nulos en cada columna
print("Cantidad de valores nulos en cada columna:")
print(null_counts.shape[0])

# Determinar si la cantidad de valores nulos es significativa
total_rows = len(df_2019)
for column, null_count in null_counts.items():
  if null_count > 0:
    percentage_null = (null_count / total_rows) * 100
    if percentage_null > 5:
      print(f"La columna '{column}' tiene una cantidad significativa de valores nulos ({percentage_null:.2f}%).")
    else:
      print(f"La columna '{column}' tiene una cantidad no significativa de valores nulos ({percentage_null:.2f}%).")
```

Algunos de los resultados obtenidos fueron los siguientes

```
La columna 'tiene_hijos' tiene una cantidad significativa de valores nulos (5.07%).
La columna 'habitaciones_en_casa' tiene una cantidad no significativa de valores nulos (4.34%).
La columna 'tiene_auto' tiene una cantidad significativa de valores nulos (7.27%).
La columna 'tiene_baño' tiene una cantidad significativa de valores nulos (5.59%).
La columna 'tiene_computadora' tiene una cantidad significativa de valores nulos (7.05%).
La columna 'tiene_tablet' tiene una cantidad significativa de valores nulos (8.36%).
La columna 'tiene_internet' tiene una cantidad significativa de valores nulos (7.80%).
La columna 'celular_propio' tiene una cantidad no significativa de valores nulos (3.88%).
La columna 'celular_propio_internet' tiene una cantidad significativa de valores nulos (8.15%).
La columna 'libros_casa' tiene una cantidad no significativa de valores nulos (4.12%).
La columna 'nivel_educativo_madre' tiene una cantidad no significativa de valores nulos (4.44%).
La columna 'nivel_educativo_padre' tiene una cantidad significativa de valores nulos (5.34%).
La columna 'cuida_familiar' tiene una cantidad significativa de valores nulos (12.87%).
La columna 'realiza_tareas_hogar' tiene una cantidad significativa de valores nulos (7.29%).
La columna 'ayuda_trabajo_padres' tiene una cantidad no significativa de valores nulos (4.60%).
La columna 'dias_trabajo_fuera_casa' tiene una cantidad significativa de valores nulos (8.20%).
La columna 'trabaja_remunerado' tiene una cantidad significativa de valores nulos (8.37%).
La columna 'fuera_escuela_junto_amigos' tiene una cantidad significativa de valores nulos (6.20%).
```

Generar Csv para cada año

En base al dataframe obtenido, generamos un csv

```python
# Guardar el DataFrame en un archivo CSV
df_2019.to_csv('Dataset_2019_preprocesado_para_union.csv', index=False)

# Descargar el archivo CSV (opcional)
files.download('Dataset_2019_preprocesado_para_union.csv')
```

Creación de dataframe en base a los archivos de 2019 y 2022

Cargamos cada uno de los dataframes en un nuevo colab y luego los concatenamos.

```
#Invocación del archivo csv de 2022
url_2022 = 'https://drive.google.com/uc?export=download&id=1hMBtdh9qgGi80iSvR9F3FHSI2vxKxfIC'
output_2022 = 'data_2022.csv'
gdown.download(url_2022, output_2022, quiet=False)

#Carga del archivo csv al dataframe df_2022
df_2022 = pd.read_csv(output_2022,sep=',')
df_2022.head(10)
```

```
#Invocación del archivo csv de 2019
url_2019 = 'https://drive.google.com/uc?export=download&id=1oJJZc2PgFHEDtG9Bb9M5grLJ69Aq4A4d'
output_2019 = 'data_2019.csv'
gdown.download(url_2022, output_2019, quiet=False)

#Carga del archivo csv al dataframe df_2019
df_2019 = pd.read_csv(output_2019,sep=',')
df_2019.head(10)
```

```
# prompt: unificar los dataframe de 2019 y 2022

import pandas as pd
df_unificado = pd.concat([df_2019, df_2022], ignore_index=True)
df_unificado.head(10)
```

Análisis para eliminar columnas

Analizamos las columnas de los dataframe unificados, eliminando las columnas que consideramos que no son necesarias.

Id1, Id Sección, Id alumno:

Las columnas referidas a algún Id, en este caso, consideramos que no son necesarias, ya que no aportan información, así que las eliminamos.

```
df_unificado = df_unificado.drop(['ID1', 'idseccion', 'idalumno'], axis=1)
df_unificado.head(10)
```

Determinamos cuales son las columnas del dataset unificado que tienen una cantidad significativa de nulls.

```python
# Contar la cantidad de valores nulos en cada columna
null_counts = df_unificado.isnull().sum()

# Mostrar la cantidad de valores nulos en cada columna
print("Cantidad de valores nulos en cada columna:")
print(null_counts)

# Determinar si la cantidad de valores nulos es significativa
total_rows = len(df_unificado)
columns_with_significant_nulls = []
for column, null_count in null_counts.items():
  if null_count > 0:
    percentage_null = (null_count / total_rows) * 100
    if percentage_null > 10:
      columns_with_significant_nulls.append(column)

print("\nColumnas con una cantidad significativa de datos nulos:")
print(columns_with_significant_nulls)
```

Las columnas con una cantidad significativa de nulos fueron:

```
['nivel_educativo_padre', 'repitio_primaria', 'repitio_secundaria_1',

 'repitio_secundaria_2', 'enseñar_sobre_cambio_climatico',

'enseñar_sobre_conflictos_internacionales', 'enseñar_sobre_salud',

'enseñar_sobre_uso_tecnologias', 'enseñar_sobre_demanda_laboral']
```

Enseñar sobre uso de tecnologías, enseñar sobre el cambio climático, enseñar sobre salud, enseñar sobre conflictos internacionales, enseñar sobre demanda laboral:

Las columnas que tienen que ver con “enseñar sobre” se refiere a si los estudiantes creen que el colegio debería enseñar estos temas, lo cual no afecta a nuestro objetivo, así que también las eliminamos.

```
df_unificado = df_unificado.drop(['enseñar_sobre_uso_tecnologias',
                                  'enseñar_sobre_cambio_climatico',
                                  'enseñar_sobre_salud',
                                  'enseñar_sobre_conflictos_internacionales',
                                  'enseñar_sobre_demanda_laboral'], axis=1)
df_unificado.head(10)
```

Hay acuerdos de convivencia, el alumno conoce sobre los acuerdos, participan en la definición:

Eliminamos las columnas relacionadas con los acuerdos de convivencia porque la gran mayoría de estudiantes o bien no tienen nada de eso en el colegio o lo desconocen.

```
df_unificado = df_unificado.drop(['acuerdo_convivencia_participacion_alumnos_definicion',
                                  'alumno_conoce_acuerdo_convivencia',
                                  'hay_acuerdo_convivencia'], axis=1)
df_unificado.head(10)
```

Repitió primaria, repitió algunos de los primeros tres años(1°, 2° o 3°) de secundaria, repitió algunos de los últimos años (4°, 5° o 6°) de secundaria:

Antes de eliminar las columnas 'repitio_primaria', 'repitio_secundaria_1', 'repitio_secundaria_2' determinamos si la gran mayoría de estudiantes NO habían repetido si ese era el caso, las eliminábamos.

```
# prompt: determinar si la gran mayoria de estudiantes no repitio primaria (valor 0 en la columna repitio_primaria)

# Calcula la proporción de estudiantes que NO repitieron primaria (valor 0 en 'repitio_primaria')
proporcion_no_repitio_primaria = (df_unificado['repitio_primaria'] == 0).mean()

# Imprime la proporción
print(f"Proporción de estudiantes que NO repitieron primaria: {proporcion_no_repitio_primaria:.2f}")

# Determina si la gran mayoría de los estudiantes NO repitieron primaria
if proporcion_no_repitio_primaria > 0.5:
  print("La gran mayoría de los estudiantes NO repitieron primaria.")
else:
  print("La gran mayoría de los estudiantes SI repitieron primaria.")
```

```
Proporción de estudiantes que NO repitieron primaria: 0.81
La gran mayoría de los estudiantes NO repitieron primaria.
```

Hicimos lo mismo para la secundaria obteniendo los siguientes resultados.

```
Proporción de estudiantes que NO repitieron secundaria_1: 0.78
La gran mayoría de los estudiantes NO repitieron secundaria_1.
Proporción de estudiantes que NO repitieron secundaria_2: 0.80
La gran mayoría de los estudiantes NO repitieron secundaria_2.
```

Como la gran mayoría de estudiantes NO repitieron y esas columnas tienen una cantidad significativa de datos nulls, entonces, eliminamos esas columnas.

Sufrió bullying:
Analizamos la columna sobre bullying para determinar si la gran mayoría de estudiantes sufrió bullying o no y decidimos eliminarla.

```python
# Calcula la proporción de estudiantes que sufrieron bullying en el aspecto físico (valor 1 en 'bullying_aspecto_fisico')
proporcion_bullying_aspecto_fisico = (df_unificado['bullying_aspecto_fisico'] == 1).mean()

# Imprime la proporción
print(f"Proporción de estudiantes que sufrieron bullying en el aspecto físico: {proporcion_bullying_aspecto_fisico:.2f}")

# Determina si la gran mayoría de los estudiantes sufrieron bullying en el aspecto físico
if proporcion_bullying_aspecto_fisico > 0.5:
  print("La gran mayoría de los estudiantes sufrieron bullying en el aspecto físico.")
else:
  print("La gran mayoría de los estudiantes NO sufrieron bullying en el aspecto físico.")
```

```
Proporción de estudiantes que sufrieron bullying en el aspecto físico: 0.08
La gran mayoría de los estudiantes NO sufrieron bullying en el aspecto físico.
```

```python
df_unificado = df_unificado.drop(['bullying_aspecto_fisico'], axis=1)
df_unificado.head(10)
```

Eliminar lo Valores Null

```python
# Eliminar filas con valores nulos
df_unificado = df_unificado.dropna()

# Verificar si hay valores nulos restantes
print("Cantidad de valores nulos en cada columna después de eliminarlos:")
print(df_unificado.isnull().sum())
```

Análisis de valores atípicos

La mayoría de las columnas son respuestas a preguntas múltiple opción, los valores están dentro de un rango limitado, e incluso algunos ya están

codificados con un OneHot donde cada columna representa el contenido de una de las opciones a la pregunta y tienen valores 0 o 1 indicando si se selecciono la opcion o no. Por este motivo, no fue necesario aplicar modificación alguna de los datos por desviaciones ya que no se encontraron valores que estén fuera del rango de valores de las opciones.

Columnas finales

Nos quedan las siguientes 55 columnas:

cod_provincia
sector
ambito
sexo
nacionalidad_alumno
padres_extranjeros
personas_en_casa
vive_madre
vive_padre
vive_hijo
vive_tio
vive_abuelo
vive_pareja
vive_amigos
vive_otro
tiene_hijos
habitaciones_en_casa
tiene_baño
tiene_computadora
tiene_tablet
tiene_auto
tiene_internet
celular_propio
celular_propio_internet
libros_casa
nivel_educativo_madre
nivel_educativo_padre
realiza_tareas_hogar
cuida_familiar

trabaja_remunerado
fuera_escuela_deporte
fuera_escuela_clases_artes
fuera_escuela_idioma
fuera_escuela_television
asistio_jardin
faltas
motivo_faltas_problemas_clima
motivo_faltas_ganas
motivo_faltas_ayuda_casa
motivo_faltas_trabajo
motivo_faltas_otros
conflicto_tratado_involucrados
conflicto_tratado_sanciones
conflicto_trado_adultos
conflicto_tratado_equipo_orientacion
conflicto_tratado_dejarlo_pasar
lpuntaje
mpuntaje
edad
vive_hermano
dias_trabajo_fuera_casa
ayuda_trabajo_padres
fuera_escuela_junto_amigos
fuera_escuela_leer
motivo_faltas_enfermedad

Cantidad de filas finales

Determinamos la cantidad de filas del dataframe luego de todas las eliminaciones anteriores:

```
num_rows = len(df_unificado)
print(f"El DataFrame tiene {num_rows} filas.")
```

```
El DataFrame tiene 450238 filas.
```

Conclusión

Luego de un largo proceso de selección y análisis de datos, conseguimos un set de datos unificado que cumple con los requisitos para ser utilizado en un modelo de IA. La muestra cuenta con más de 400000 filas y conserva la mayoría de las columnas comunes de 2019 y 2022, que pueden ser útiles para predecir las notas de los estudiantes.