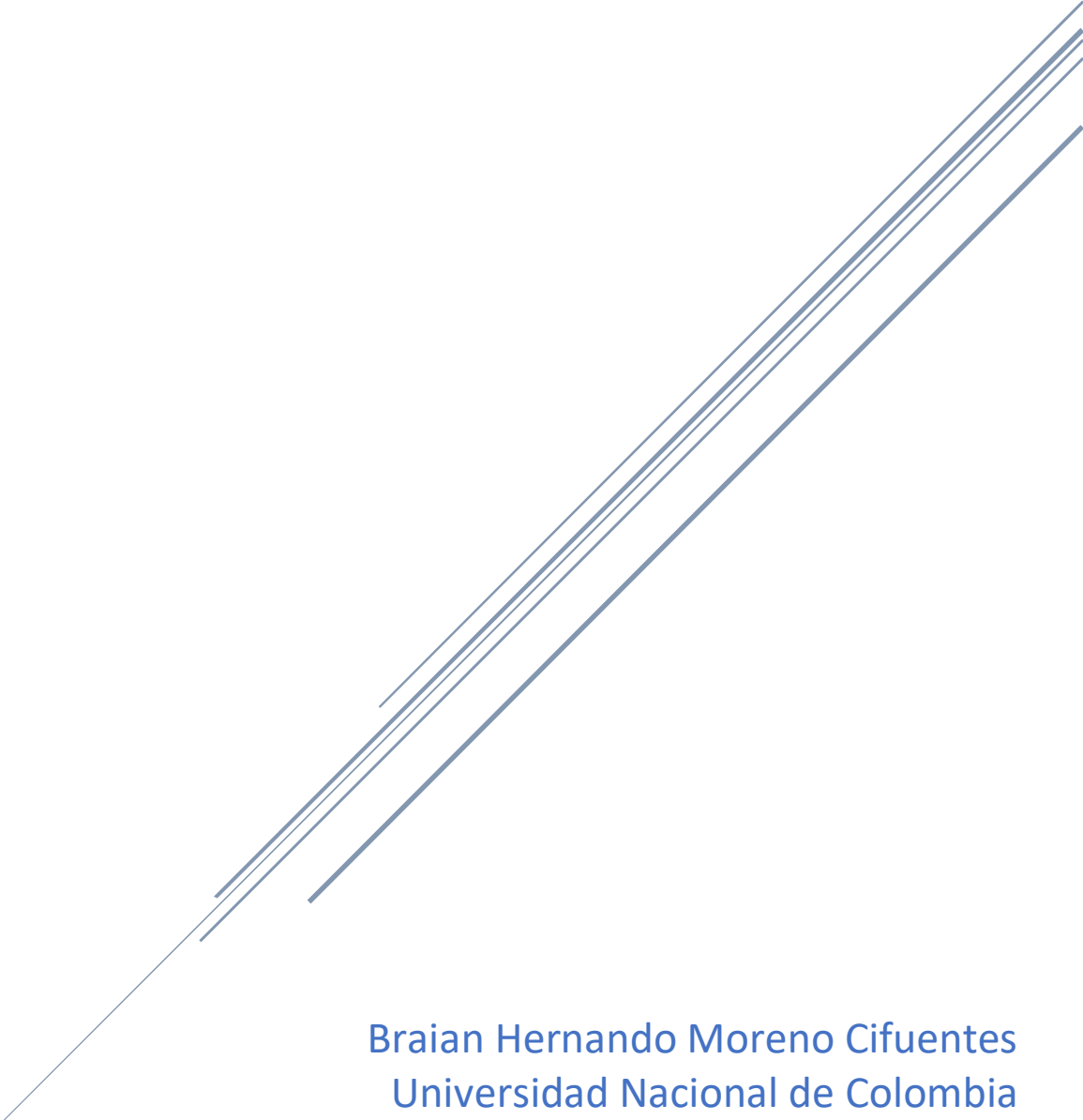


FORMULACIÓN DE RESERVAS PARA LAS ASEGURADORAS DEL SECTOR SALUD

Modelo CRISP-DM



Braian Hernando Moreno Cifuentes
Universidad Nacional de Colombia
Maestría en Actuaría y Finanzas
Aplicación del aprendizaje de máquina en Actuaría y Finanzas

Tabla de contenido

0.	INTRODUCCIÓN Y PRELIMINARES	3
	OBJETIVO	3
	ALCANCE	3
	IMPORTANCIA DE LA SOLVENCIA	3
1.	<i>Fase 1: Comprensión del negocio</i>	4
2.	<i>Fase 2: Comprensión de los datos</i>	5
	i) Recopilación inicial de los datos.....	5
	ii) Exploración de los datos.....	6
	iii) Evaluación de la calidad de los datos	10
3.	<i>Fase 3: PREPARACIÓN DE LOS DATOS (DATA PREPARATION)</i>	11
	i) Seleccionar datos relevantes	11
	ii) Limpieza y preprocesamiento de los datos	12
	iii) Transformar los datos en un formato adecuado para el modelado.....	13
4.	<i>Fase 4: Modelado (Modeling):</i>	14
	i) Construcción y entrenamiento del modelo utilizando datos de entrenamiento.....	14
	ii) Ajuste del modelo para mejorar el rendimiento	16
5.	<i>Fase 5: Evaluación (Evaluation):</i>	19
	i) Evaluación del rendimiento del modelo utilizando datos de prueba.	19
	ii) Validación del modelo con conjuntos de datos independientes.	20
	iii) Conclusión acerca de los objetivos comerciales se han alcanzado.	22

0. INTRODUCCIÓN Y PRELIMINARES

OBJETIVO

En este documento se pretende analizar el modelo Chain-Ladder para la predicción de reservas enfocadas en reservas de salud. En el negocio de los seguros, el desarrollo de las reservas es un proceso crucial para abordar, medir y definir la solvencia de una empresa para los próximos trimestres y años. La solvencia, una preocupación reglamentaria, influye directamente en la reputación, la exposición a sanciones y el éxito financiero. El concepto fundamental de la constitución de reservas implica la asignación de efectivo para cubrir posibles siniestros, por lo que la disponibilidad de efectivo es crítica para las operaciones de la compañía de seguros.

ALCANCE

Este documento abordará el problema de la Escalera en Cadena dentro del marco CRISP-DM. El análisis implicará la comprensión de los datos existentes obtenidos del sitio web de la CAS, la preparación de los datos para su validación en diferentes pruebas de modelos (determinista, regresión lineal) y, en última instancia, la realización de un análisis de validación cruzada para determinar el modelo con el menor error de predicción, lo que indica un mejor rendimiento.

IMPORTANCIA DE LA SOLVENCIA

Un sistema de reservas sólido es fundamental para una compañía de seguros, ya que garantiza la disponibilidad de fondos para hacer frente a posibles pagos de siniestros. El éxito en la definición de la solvencia no sólo cumple los requisitos reglamentarios, sino que también salvaguarda la reputación de la empresa y minimiza la exposición a sanciones. El éxito financiero de la empresa está íntimamente ligado a la eficacia de la gestión de sus reservas. Al realizar un análisis exhaustivo del modelo Chain Ladder y emplear un marco estructurado, este documento pretende mejorar la comprensión del proceso de predicción de reservas y contribuir a la gestión global del riesgo y la estabilidad financiera de la empresa, especialmente las relacionadas con el área de la salud, que tiene una gran importancia para cualquier sistema.

1. Fase 1: Comprensión del negocio

La primera fase que corresponde a la comprensión del negocio (Business Understanding) se encuentra descrita en la página personal de github, cuya dirección es: [https://github.com/Braian8825/CRISP-DM/blob/main/Fase 1 CRISP DM.ipynb](https://github.com/Braian8825/CRISP-DM/blob/main/Fase%201%20CRISP%20DM.ipynb).

Allí se muestra todo lo correspondiente al Business Understanding y se crea un cronograma con las actividades para realizar.

2. Fase 2: Comprensión de los datos.

Este documento contiene los resultados obtenidos tras realizar el análisis de comprensión y preparación de datos al fichero `medmal_pos.csv`, que contiene la información de siniestros de Negligencias Médicas LoB. El análisis se ha realizado en google collab utilizando el lenguaje de programación python en el siguiente enlace:

<https://colab.research.google.com/drive/1BQNHaoGjxOk1KvapbNldAoJaQ0DHzfnX?authuser=1>

i) Recopilación inicial de los datos

De acuerdo al paso de comprensión del negocio, se tiene una base de datos que consiste en información de reclamos de aseguradoras americanas en el área de los seguros médicos por negligencia médica en el sistema de salud. Esta base de datos en un archivo `.csv` (cuyo nombre es `medmal_pos.csv`) y contiene las siguientes variables:

`GRCODE`: Código de la aseguradora (incluidos los grupos de aseguradoras y los aseguradores individuales).

`GRNAME`: Nombre de la aseguradora (incluidos los grupos de aseguradoras y los aseguradores individuales).

`AccidentYear`: Año del accidente (desde 1988 hasta 1997).

`DevelopmentYear`: Año de desarrollo del reclamo (desde 1988 hasta 1997).

`DevelopmentLag`: Año de retraso (calculado como: año del accidente-1987 + año de desarrollo del reclamo-1987 - 1).

`IncurLoss_F2`: Pérdidas incurridas y gastos asignados reportados al final del año por negligencia médica.

`CumPaidLoss_F2`: Pérdidas pagadas acumuladas y gastos asignados al final del año por negligencia médica.

`BulkLoss_F2`: Seguro del portafolio (Bulk) y reserva de siniestros ocurridos y no reportados (IBNR reserves) sobre pérdidas netas y gastos de defensa y contención

de costos reportados al final del año por negligencia médica.

PostedReserve97_F2: Reservas contabilizadas en el año 1997 tomadas del Anexo de Suscripción e Inversiones, incluidas las pérdidas netas no pagadas y los gastos de ajuste de pérdidas no pagados por negligencia médica.

EarnedPremDIR_F2: Primas devengadas en el año incurrido: directas y asumidas por negligencia médica.

EarnedPremCeded_F2: Primas devengadas en el año incurrido: cedidas por negligencia médica.

EarnedPremNet_: Primas netas devengadas en el año en que se incurrió por negligencia médica: 1 indica una sola entidad, 0 indica una aseguradora grupal.

F2: Reclamos hechos por negligencia médica.

ii) Exploración de los datos

La base de datos que vamos a trabajar fue tomada de la webpage de Casualty Actuarial Society (CAS) cuyo website es: <https://www.casact.org/publications-research/research/research-resources/loss-reserving-data-pulled-naic-schedule-p>.

Estos archivos son públicos y no tienen ninguna restricción sobre su uso. Este formato se encuentra completo y no tiene problemas con el tratamiento de los datos. En nuestro trabajo, se inicia con la exploración inicial de los datos descritos en 1. y se calculan algunas medidas de tendencia central para corroborar si los datos son acordes o contienen outliers o valores que puedan causar problemas:

```

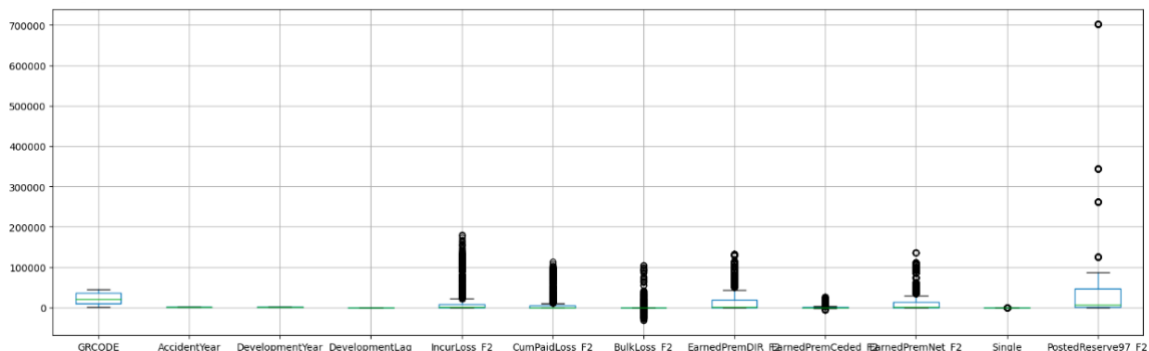
Data columns (total 13 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   GRCODE                                3400 non-null   int64
1   GRNAME                                3400 non-null   object
2   AccidentYear                          3400 non-null   int64
3   DevelopmentYear                       3400 non-null   int64
4   DevelopmentLag                        3400 non-null   int64
5   IncurLoss_F2                          3400 non-null   int64
6   CumPaidLoss_F2                       3400 non-null   int64
7   BulkLoss_F2                          3400 non-null   int64
8   EarnedPremDIR_F2                     3400 non-null   int64
9   EarnedPremCeded_F2                   3400 non-null   int64
10  EarnedPremNet_F2                      3400 non-null   int64
11  Single                                3400 non-null   int64
12  PostedReserve97_F2                   3400 non-null   int64
dtypes: int64(12), object(1)
memory usage: 345.4+ KB

```

Se observa que hay un total de 13 variables, de donde solo la variable GRNAME tiene datos cualitativos. Las demás variables manejan datos cuantitativos. Esto sucede, pues GRNAME tiene el nombre de las aseguradoras o grupo de aseguradoras que le hicieron el reclamo por negligencia médica. Los datos se describen a continuación:

	GRCODE	AccidentYear	DevelopmentYear	DevelopmentLag	IncurLoss_F2	CumPaidLoss_F2	BulkLoss_F2	EarnedPremDIR_F2	EarnedPremCeded_F2	EarnedPremNet_F2	Single	PostedReserve97_F2
count	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000	3400.000000
mean	22809.764706	1992.500000	1997.000000	5.500000	11609.344412	6706.067059	1095.803235	14111.605882	1803.497059	12308.108824	0.852941	57065.529412
std	14708.377001	2.872704	4.062617	2.872704	26802.819463	17121.815066	7612.672277	26399.284476	3893.424584	24824.225795	0.354217	134355.533990
min	669.000000	1988.000000	1988.000000	1.000000	-17.000000	-1190.000000	-32101.000000	-781.000000	-6214.000000	-728.000000	0.000000	0.000000
25%	10341.000000	1990.000000	1994.000000	3.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	629.000000
50%	19764.000000	1992.500000	1997.000000	5.500000	645.000000	187.000000	0.000000	1500.000000	106.500000	1302.000000	1.000000	5875.000000
75%	36234.000000	1995.000000	2000.000000	8.000000	9050.500000	4385.500000	107.250000	18094.500000	1473.500000	13490.000000	1.000000	46762.000000
max	44504.000000	1997.000000	2006.000000	10.000000	179425.000000	113189.000000	104402.000000	131948.000000	25553.000000	135318.000000	1.000000	702246.000000

Los respectivos outliers están dados por:



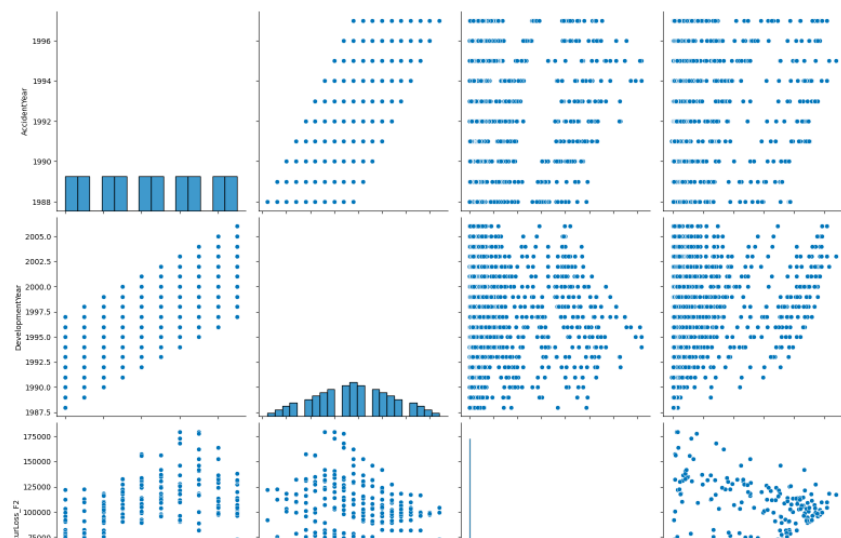
Se puede apreciar que varias variables contienen datos atípicos. Es bastante complicado en una primera vez pensar en modelar datos con tantos datos atípicos. Es así como podemos entender el proceso que debe tener la codificación detrás para tener éxito. Por otro lado, los registros que se tienen por empresas son:

```

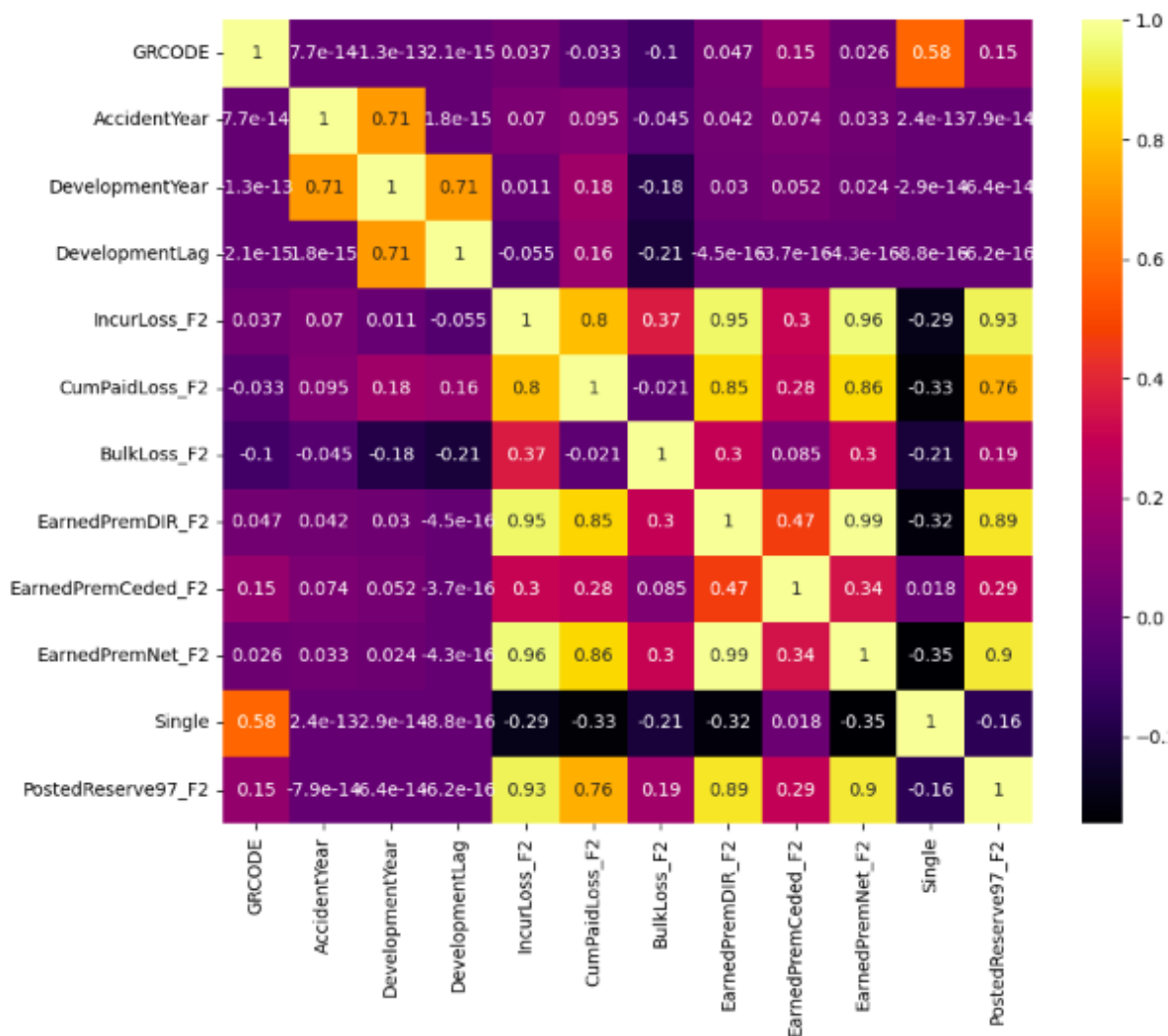
Scpie Indemnity Co          100
Preferred Professional Ins Co 100
Nichido Fire & Marine Ins Co Ltd 100
Texas Hospital Ins Exch    100
State Volunteer Mut Ins Co 100
MHA Ins Co                 100
Health Care Ind Inc        100
National Guardian RRG Inc  100
Medical Mut Ins Co Of ME   100
Promutual Grp              100
Utah Medical Ins Assoc     100
Seguros Triples Inc        100
Dentists Ins Co            100
Physicians Recip Insurers  100
Louisiana Med Mut Ins Co   100
Clinic Mut Ins Co RRG      100
Michigan Professional Ins Exch 100
National American Ins Co   100
NCMIC Ins Co               100
Underwriters At Lloyds London 100
Community Blood Cntr Exch RRG 100
Campmed Cas & Ind Co Inc MD 100
Homestead Ins Co           100
Franklin Cas Ins Co RRG    100
MCIC VT Inc RRG            100
Texas Medical Ins Co       100
Controlled Risk Ins Co Of VT Inc 100
American Assoc Of Othodontists RRG 100
Eastern Dentists Ins Co RRG 100
Overseas Partners Us Reins Co 100
Markel Corp Grp            100
Nationwide Grp             100
Great Amer Grp             100
California Healthcare Ins Co Inc 100
Name: GRNAME, dtype: int64

```

Sacando analítica descriptiva básica de las variables de la base de datos correspondiente a cuantas variables tiene, la media de cada una de los datos, desviaciones estándar entre otras.



Las correlaciones que genera la base de datos muestra las correlaciones correspondientes entre cada una de las variables. Nos interesa observar si existe alguna relación entre las variables. Se intuye que si la variable se compara con ella misma, su correlación deberá ser un valor igual a 1. Esto nos servirá para ver si el modelo que vamos a construir se puede obtener con las variables correlacionadas. Si es posible, los resultados podrían servir para empresas que tengan la misma estructura:



Vemos que se forma una matriz que muestra las correlaciones correspondientes entre todas las variables. Es claro que es 1 si una variable está correlacionada con ella misma: Por ejemplo,

AccidentYear tiene una correlación de 1 con ella misma, pero tiene una correlación de 0,18 con CumPaidLoss_F2.

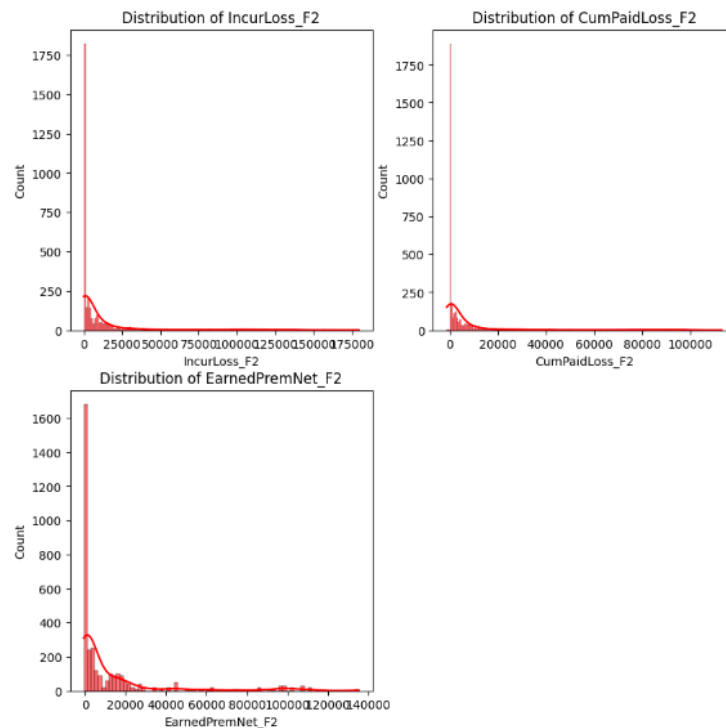
iii) Evaluación de la calidad de los datos

Al obtener los datos de una página actuarial que brinda confiabilidad en ellos y que ha sido trabajada previamente, se concluye que los datos tienen excelente calidad y por ende no se les debe hacer ninguna modificación ni ningún ajuste. Se menciona que éstos son verídicos y confiables. Se puede consultar en el link previo. Por ende, se dice que la calidad de los datos es del 100%.

3. Fase 3: PREPARACIÓN DE LOS DATOS (DATA PREPARATION)

i) Seleccionar datos relevantes.

Una vez vista la calidad de los datos, características y evaluar junto al equipo de trabajo y los interesados en que el modelo funcione, los datos a trabajar son: 'AccidentYear', 'IncurLoss_F2', 'CumPaidLoss_F2' y 'EarnedPremNet_F2', pues se considera que con ellas podemos calcular lo que necesitamos. Tenga en cuenta que buscamos la manera óptima de tratar un problema de minería de datos donde si usamos todos los parámetros tendremos sobreajuste (overfitting) y el modelo no sería óptimo. Para los análisis descriptivos, no se considerará la variable 'AccidentYear' por el tipo de variable (categórica) que es diferente al tipo de las otras variables (numéricas). En este caso, las gráficas de distribución son, respectivamente:



teniendo en cuenta nuestro objetivo que es realizar un trabajo para estimar por método Chain Ladder (método estadístico utilizado para el cálculo de la provisión de prestaciones, basado en el análisis de los triángulos de siniestros) y los triángulo de siniestros corresponden a una distribución bidimensional de la

información histórica de siniestralidad. generalmente las dos dimensiones son el año de ocurrencia (eje vertical) y el año de pago (eje horizontal). También, debemos tener en cuenta que tengo que hacer el modelo con cierta cantidad de datos y reservar otros datos para testear el modelo y ver si se puede generalizar con datos externos a la muestra inicial por lo mismo lo primero es que se va a reservar los datos de una compañía para las pruebas el modelo y el resto para entrenamiento.

Triángulo de siniestros usando Chain Ladder.

Este triángulo es una herramienta gráfica y analítica que se utiliza para realizar un seguimiento y analizar la evolución de las pérdidas o siniestros a lo largo del tiempo. Cada vértice del triángulo representa un año o período, y los datos en el triángulo muestran cómo las pérdidas se desarrollan y cambian a medida que pasa el tiempo. Este se utiliza en la actuaria para estimar la cantidad de siniestros pendientes (siniestros que aún no se han liquidado por completo) y para proyectar futuros siniestros y costos relacionados con seguros. Los actuarios analizan estos triángulos para evaluar la siniestralidad, establecer reservas y tomar decisiones sobre la fijación de primas de seguros, entre otras aplicaciones. Para nuestra aplicación, seleccionamos las variables de interés y les hacemos la respectiva limpieza de los datos, la eliminación de los ceros si éstos existen y su posterior resultado.

ii) Limpieza y preprocesamiento de los datos.

En este paso, mostramos algunos pasos que se implementaron en el código que se ha construido. Para total consulta, puede remitirse a la web Page de Python usando colab:

<https://colab.research.google.com/drive/1BQNHOGjxOk1KvapbNldAoJaQ0DHZfnX?authuser=1>

Los valores de entrenamiento y prueba están dados por el código:

```
df_train = df.head(3300).copy()
df_train['GRNAME'].value_counts()
```

Aquí, se está tomando las primeras 3300 filas del DataFrame original y se crea un nuevo DataFrame llamado `df_train`. El método `head(3300)` selecciona las primeras 3300 filas, y luego se usa `copy()` para crear una copia independiente del DataFrame. Esto es importante porque asegura que `df_train` sea una copia y no simplemente una referencia al mismo objeto que `df` cuidando los datos originales. También, es necesario hacerlo porque el total de datos faltantes se utilizará para la parte de test del modelo y éstos no se pueden tocar. También, se accede a la columna 'GRNAME' del DataFrame `df_train` y se utiliza el método `value_counts()` para contar la frecuencia de cada valor único en esa columna. En otras palabras, esta línea de código mostrará cuántas veces aparece cada valor único en la columna 'GRNAME' del DataFrame `df_train`. Esto puede ser útil para entender la distribución de los valores en esa columna.

- iii) Transformar los datos en un formato adecuado para el modelado.

Después de seleccionar las filas y columnas para utilizar, retirar de la muestra los elementos que presentan 0 en los datos para evitar sesgos en la muestra se acaba de validar que se eliminaron las compañías que tenía datos de 0 y ya tenemos la base de datos limpia para poder correr nuestra chain ladder.

4. Fase 4: Modelado (Modeling):

La construcción del modelo Chain Ladder es de mucha utilidad en el mundo de la actuaría ya que este método es el utilizado para realizar los triángulos de estimación de reservas. La idea consiste en revisar método diferentes y comparar los resultados existentes.

i) Construcción y entrenamiento del modelo utilizando datos de prueba (training)

La construcción del modelo es de la siguiente forma:

```
class ChainLadder:
    def __init__(self, tabla = pd.DataFrame(), origin = "", development = "",
columns = "", index = ""):

        self.tabla = tabla #OK
        self.origin = origin #OK
        self.development = development #OK
        self.index = index #OK
        self.columns = columns #OK

    def Triangulos(self):

        # Renombrar las columnas
        datos = self.tabla.rename(columns={self.origin: "AccidentYear",
self.development: "DevelopmentLag", self.columns: "IncurLoss_C",
self.index: "GRCODE"})

        diccionario_todos_triangulos = {}

        for k in datos["GRCODE"].unique():

            Filtro_datos = datos[datos["GRCODE"] == k]

            Triangulo_full = Filtro_datos.pivot_table(values = "IncurLoss_C",
index = "AccidentYear", columns='DevelopmentLag', aggfunc="sum", margins=False)

            Triangulo_full_acumulado = Triangulo_full.copy()

            num_filas = Triangulo_full_acumulado.shape[0]
            num_columnas = Triangulo_full_acumulado.shape[1]

            Triangulo_full_mitad = Triangulo_full.copy()
            for i in range(num_filas):
                for j in range(1,i+1):
                    Triangulo_full_mitad.iloc[i, -j] = None # Puedes establecerlo
en None u otro valor si lo prefieres

            for indice, i in enumerate(range(1,num_columnas+1)):
                Triangulo_full_acumulado[Triangulo_full.columns[indice]] =
Triangulo_full[Triangulo_full.columns[0:i]].sum(axis = 1)

            Triangulo_acumulado_mitad = Triangulo_full_acumulado.copy()
            for i in range(num_filas):
                for j in range(1,i+1):
                    Triangulo_acumulado_mitad.iloc[i, -j] = None # Puedes
establecerlo en None u otro valor si lo prefieres

            factores0 = Triangulo_acumulado_mitad.sum(axis = 0) # Rojo
```

```

        factores1 = Triangulo_acumulado_mitad.sum(axis = 0) -
np.flip(np.diag(np.fliplr(Triangulo_acumulado_mitad), 0)) # Azul
        factores0 = factores0[1:10]
        factores1 = factores1[0:-1]
        factores = factores0.reset_index(drop = True) /
factores1.reset_index(drop = True)

        Triangulo_estimado = Triangulo_acumulado_mitad.copy()
        for i in list(reversed(range(num_filas))):
            comodin = np.diag(np.fliplr(Triangulo_acumulado_mitad), 0)[i]
            for j in range(1,i+1):
                Triangulo_estimado.iloc[i, -j] = comodin*factores.iloc[-i+9:-
j+10].prod() # Puedes establecerlo en None u otro valor si lo prefieres

        reserva_total =
sum(np.array(list(reversed(np.array(Triangulo_estimado[10])))))-
np.flip(np.diag(np.fliplr(Triangulo_estimado), 0)))

        diccionario_triangulo = {'Triangulo_full':Triangulo_full,
"Triangulo_full_mitad":Triangulo_full_mitad,
"Triangulo_full_acumulado":Triangulo_full_acumulado,

"Triangulo_acumulado_mitad":Triangulo_acumulado_mitad, "factores":factores,
"Triangulo_estimado":Triangulo_estimado,
"reserva_total":reserva_total}

        nombre = k
        diccionario_todos_triangulos[nombre] = diccionario_triangulo

    return diccionario_todos_triangulos

```

Donde este método se llama cuando se crea un nuevo objeto de la clase ChainLadder. Los parámetros son opciones para inicializar las propiedades de la clase (tabla, origin, development, columns, index). Estos parámetros son utilizados para definir atributos de la instancia, que parecen representar nombres de columnas en un DataFrame.

Con respecto a Triangulos, se busca realizar el cálculo de triángulos de desarrollo y estimación de reservas para diferentes categorías de riesgo. Crea un diccionario llamado diccionario_todos_triangulos para almacenar los resultados de cada categoría de riesgo. Utiliza un bucle for para iterar sobre valores únicos en la columna "GRCODE" del DataFrame y realiza cálculos para cada categoría de riesgo. Dentro del bucle for: Se realizan operaciones de manipulación y transformación de datos para obtener triángulos acumulativos y estimaciones de reservas. Los resultados de cada paso se almacenan en el diccionario diccionario_triangulo. Este diccionario se agrega al diccionario general diccionario_todos_triangulos con la clave siendo el valor único de "GRCODE".

Finalmente, El método descrito anteriormente devuelve `diccionario_todos_triangulos`, que contiene los resultados del cálculo de triángulos para cada categoría de riesgo. El resultado es el siguiente:

DevelopmentLag	1	2	3	4	5	6	7	8	9	10
AccidentYear										
1988	14401	14191.0	12843.0	9472.0	9965.0	8242.0	8089.0	7488.0	8116.0	7729.0
1989	13744	12906.0	10525.0	9371.0	7871.0	7566.0	7110.0	7148.0	6561.0	NaN
1990	15949	16354.0	15852.0	13053.0	11737.0	11019.0	11176.0	9656.0	NaN	NaN
1991	18942	16006.0	13405.0	11538.0	12349.0	12271.0	10885.0	NaN	NaN	NaN
1992	20949	20157.0	17870.0	16704.0	16469.0	14768.0	NaN	NaN	NaN	NaN
1993	21684	20708.0	17483.0	16437.0	14440.0	NaN	NaN	NaN	NaN	NaN
1994	19198	18597.0	17124.0	14480.0	NaN	NaN	NaN	NaN	NaN	NaN
1995	21489	21409.0	20066.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1996	20427	18211.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1997	18961	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Se observa un triángulo predictivo desde el año 1988 a un máximo de 10 años.

ii) Ajuste del modelo para mejorar el rendimiento

Se presenta a continuación una implementación diferente con el fin de. Mejorar el rendimiento. La codificación es la siguiente:

```
class ChainLadder_corto:
    def __init__(self, tabla = pd.DataFrame(), origin = "", development = "",
columns = "", index = ""):

        self.tabla = tabla #OK
        self.origin = origin #OK
        self.development = development #OK
        self.index = index #OK
        self.columns = columns #OK

    def Triangulos(self):

        # Renombrar las columnas
        datos = self.tabla.rename(columns={self.origin: "AccidentYear",
self.development: "DevelopmentLag", self.columns: "IncurLoss_C",
self.index: "GRCODE"})

        diccionario_todos_triangulos = {}

        for k in datos["GRCODE"].unique():

            Filtro_datos = datos[datos["GRCODE"] == k]
```



```

        Triangulo_full = Filtro_datos.pivot_table(values = "IncurLoss_C",
index = "AccidentYear", columns='DevelopmentLag', aggfunc="sum", margins=False)

        Triangulo_full_acumulado = Triangulo_full.copy()

        num_filas = Triangulo_full_acumulado.shape[0]
        num_columnas = Triangulo_full_acumulado.shape[1]

        Triangulo_full_mitad = Triangulo_full.copy()
        for i in range(num_filas):
            for j in range(1,i+1):
                Triangulo_full_mitad.iloc[i, -j] = None # Puedes establecerlo
en None u otro valor si lo prefieres

        diccionario_triangulo = {'Triangulo_full':Triangulo_full,
"Triangulo_full_mitad":Triangulo_full_mitad}

        nombre = k
        diccionario_todos_triangulos[nombre] = diccionario_triangulo

    return diccionario_todos_triangulos

```

Este método lo nombramos Chain Ladder corto y consiste en que los parámetros son opciones para inicializar las propiedades de la clase (tabla, origin, development, columns, index). Estos parámetros son utilizados para definir atributos de la instancia, que parecen representar nombres de columnas en un DataFrame. Por otro lado, el método Triangulos(self) realiza el cálculo de triángulos de desarrollo para diferentes categorías de riesgo. Crea un diccionario llamado diccionario_todos_triangulos para almacenar los resultados de cada categoría de riesgo. Utiliza un bucle for para iterar sobre valores únicos en la columna "GRCODE" del DataFrame y realiza cálculos para cada categoría de riesgo. Dentro del bucle for, se realizan operaciones de manipulación y transformación de datos para obtener triángulos acumulativos y estimaciones de reservas. Los resultados de cada paso se almacenan en el diccionario diccionario_triangulo. Este diccionario se agrega al diccionario general diccionario_todos_triangulos con la clave siendo el valor único de "GRCODE". Finalmente: El método devuelve diccionario_todos_triangulos, que contiene los resultados del cálculo de triángulos para cada categoría de riesgo.

Al validar los códigos de las compañías que aún tenemos en la base de datos, se tiene el siguiente triángulo:

DevelopmentLag	1	2	3	4	5	6	7	8	9	10
AccidentYear										
1988	121905	112211.0	103226.0	99599.0	96006.0	90487.0	82640.0	80406.0	78920.0	78511.0
1989	122679	113165.0	110037.0	101142.0	90817.0	81919.0	77491.0	73577.0	72716.0	NaN
1990	118157	117497.0	116377.0	99895.0	89252.0	81916.0	79134.0	76333.0	NaN	NaN
1991	117981	122443.0	121056.0	113795.0	102830.0	98071.0	94870.0	NaN	NaN	NaN
1992	131059	130155.0	124195.0	113974.0	106817.0	99182.0	NaN	NaN	NaN	NaN
1993	134700	130757.0	125253.0	114717.0	111294.0	NaN	NaN	NaN	NaN	NaN
1994	136749	128192.0	121355.0	111877.0	NaN	NaN	NaN	NaN	NaN	NaN
1995	140962	132405.0	118332.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1996	134473	128980.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1997	137944	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

De ahora en adelante, se trabajará con este triángulo por simplicidad ya que arroja resultado de manera más rápida y hace las cuentas de manera más simple. El triángulo estimado con respecto al acumulado es:

DevelopmentLag	1	2	3	4	5	6	7	8	9	10
AccidentYear										
1988	121905	234116.00000	337342.00000	436941.00000	532947.00000	623434.00000	706074.00000	786480.00000	865400.00000	9.439110e+05
1989	122679	235844.00000	345881.00000	447023.00000	537840.00000	619759.00000	697250.00000	770827.00000	843543.00000	9.200711e+05
1990	118157	235654.00000	352031.00000	451926.00000	541178.00000	623094.00000	702228.00000	778561.00000	854369.993215	9.318803e+05
1991	117981	240424.00000	361480.00000	475275.00000	578105.00000	676176.00000	771046.00000	855386.937928	938676.522910	1.023835e+06
1992	131059	261214.00000	385409.00000	499383.00000	606200.00000	705382.00000	798084.554401	885383.107099	971593.438671	1.059739e+06
1993	134700	265457.00000	390710.00000	505427.00000	616721.00000	716316.455938	810456.035945	899107.844295	986654.562494	1.076166e+06
1994	136749	264941.00000	386296.00000	498173.00000	603790.859451	701298.202217	793464.056665	880257.195792	965968.414031	1.053603e+06
1995	140962	273367.00000	391699.00000	507257.864803	614801.810119	714087.332406	807933.957043	896309.887559	983584.166882	1.072817e+06
1996	134473	263453.00000	386575.428325	500622.739291	606759.968961	704746.798910	797365.874057	884585.813839	970718.512445	1.058784e+06
1997	137944	270785.33457	397334.464575	514555.901631	623647.106690	724361.072868	819557.890711	909205.306285	997735.220954	1.088252e+06

Con lo anterior, se concluye que la reserva total que debe tener la aseguradora (predicción del valor óptimo) es 4278625.832135844 millones.

5. Fase 5: Evaluación (Evaluation):

- i) Evaluación del rendimiento del modelo utilizando datos de prueba.

Las regresiones Ridge y Lasso son técnicas de regularización utilizadas en modelos de regresión para abordar el problema de la multicolinealidad y, al mismo tiempo, para prevenir el sobreajuste. Ambos métodos introducen términos de penalización en la función de costo del modelo para limitar el tamaño de los coeficientes de las variables predictoras.

Con respecto a la regresión de Ridge, el objetivo es Minimizar la función de costo, que es la suma de los errores al cuadrado más un término de penalización proporcional al cuadrado de los coeficientes donde los efectos que presenta consiste en que los coeficientes de Ridge son "encogidos" hacia cero, pero nunca exactamente a cero. Incluso variables menos importantes contribuirán al modelo, aunque en menor medida. Es útil cuando hay multicolinealidad en los datos, es decir, cuando las variables predictoras están altamente correlacionadas entre sí.

Por otra parte, en la regresión Lasso el objetivo es minimizar la función de costo, que es la suma de los errores al cuadrado más un término de penalización proporcional al valor absoluto de los coeficientes. El efecto que produce en ellos consiste en realizar la selección de características al forzar algunos coeficientes a ser exactamente cero, eliminando así algunas variables predictoras del modelo. Es Útil cuando se sospecha que solo un subconjunto de variables predictoras es realmente relevante, lo que conduce a un modelo más simple y fácil de interpretar.

Ambos métodos controlan el sobreajuste y mejoran la estabilidad del modelo al limitar el tamaño de los coeficientes. Ridge suele ser preferido cuando todas las variables son potencialmente relevantes, mientras que Lasso puede ser preferido cuando se espera que solo un pequeño número de variables sea relevante. La elección entre Ridge y Lasso, o una combinación de ambos (Elastic Net), depende de la naturaleza específica del

problema y de la importancia relativa de las características en el modelo. La selección del parámetro de penalización λ también es crucial y se puede ajustar mediante validación cruzada. Los resultados al aplicar las regresiones al conjunto de datos son los siguientes:

```
{'Coef_normal': array([ 0.          , -0.05366319, -0.12604845, -0.22685027, -0.28797468,
-0.35457203, -0.39991772, -0.49217904, -0.49546519, -0.51181813,
 0.29862739,  0.39832096,  0.53248843,  0.60872173,  0.63867745,
 0.75294324,  0.81777038,  0.84446757,  0.69437918]),
'Coef_ridge1': array([ 0.          , -0.05364488, -0.12603211, -0.22683501, -0.28796046,
-0.35455841, -0.39990494, -0.49216591, -0.49545364, -0.51180904,
 0.29858475,  0.39827697,  0.53244259,  0.60867406,  0.63862804,
 0.75288984,  0.8177119 ,  0.84440078,  0.69430212]),
'Coef_lasso': array([ 0.          , -0.          , -0.07173795, -0.17254086, -0.23366693,
-0.30026734, -0.34561888, -0.4378919 , -0.4412045 , -0.45764135,
 0.24350776,  0.34320307,  0.47737344,  0.55360948,  0.5835672 ,
 0.69783375,  0.76265966,  0.7893516 ,  0.6385957 ]))}
```

Y la evaluación de los tres modelos en el conjunto de validación o aseguradora de validación son:

```
{'MAPE': 17.862565448352345,
'MAPE_ridge_1': 17.862732438617133,
'MAPE_lasso': 17.86437860509435}
```

En este conjunto de datos, la mejor validación la proporciona la regresión lineal; esto quiere decir que regresión lineal simple sin regularización produce un modelo que se ajusta mejor a los datos en un determinado contexto. Hay que decir que la elección entre la regresión lineal simple y los métodos de regularización depende de la complejidad del problema, la cantidad de datos disponibles, la presencia de multicolinealidad y la interpretabilidad del modelo resultante. Cada enfoque tiene sus ventajas y desventajas, y la elección debe basarse en la comprensión del problema específico y los objetivos del modelado.

- ii) Validación del modelo con conjuntos de datos independientes.

La validación cruzada es una técnica utilizada en el aprendizaje automático para evaluar el rendimiento de

un modelo estadístico y, al mismo tiempo, minimizar los problemas asociados con una única división de datos en conjuntos de entrenamiento y prueba. El objetivo es obtener una evaluación más robusta y generalizable del modelo. Uno de los métodos más comunes de validación cruzada es la validación cruzada k-fold. Los pasos para hacer la validación son:

- **División de los Datos:** Se dividen los datos disponibles en k partes o "folds" (normalmente, k se elige en el rango de 5 a 10). Cada fold se utiliza como conjunto de prueba exactamente una vez, mientras que los k-1 folds restantes se utilizan como conjunto de entrenamiento.
- **Iteración:** El modelo se entrena y evalúa k veces, cada vez utilizando un fold diferente como conjunto de prueba y los demás como conjunto de entrenamiento.
- **Promedio de Resultados:** Se promedian los resultados de las k iteraciones para obtener una medida de rendimiento más robusta y generalizable.
- **Evaluación del Rendimiento:** Se evalúa el rendimiento del modelo utilizando métricas relevantes, como precisión, recall, F1-score, MSE (Error Cuadrático Medio), etc., dependiendo del tipo de problema.

La validación cruzada es útil para ajustar parámetros del modelo mediante búsqueda de hiperparámetros. Se puede realizar una búsqueda sobre una cuadrícula de valores de hiperparámetros, y la combinación que produce el mejor rendimiento promedio a lo largo de las k iteraciones se selecciona como la mejor. La validación cruzada es esencial para obtener estimaciones más precisas del rendimiento del modelo, especialmente cuando se trabaja con conjuntos de datos pequeños o cuando la variabilidad en la elección del conjunto de entrenamiento y prueba es un problema importante.

Con respecto a la selección del mejor modelo, el código calcula el MAPE (Mean Absolute Percentage Error) para el método Chain-Ladder de estimación de reservas de seguros. El MAPE es una medida de la precisión de los modelos de previsión, y se calcula promediando los errores porcentuales absolutos de cada predicción

realizada por el modelo. En primer lugar, el código itera sobre cada aseguradora del conjunto de datos. Para cada aseguradora, calcula la diferencia absoluta entre las pérdidas estimadas y las observadas para todos los triángulos y todos los desfases de desarrollo. A continuación, divide la suma de estas diferencias por el número total de triángulos y multiplica el resultado por 100 para expresar el MAPE en porcentaje. El valor MAPE de cada asegurador se adjunta a una lista. Por último, el código calcula el MAPE medio de todas las aseguradoras e imprime el resultado. Este valor medio de MAPE representa el rendimiento global del método Chain-Ladder para el conjunto de datos dado. Y el resultado que se obtuvo fue:

Métrica MAPE con el método Chain-Ladder: 7.036457837171299
Métrica MAPE con el modelo final: 0.932000384362218

Mostrando que la regresión Lasso brinda una mejor aproximación al modelo. Comparando estos dos valores de MAPE, el modelo final (Lasso) parece tener un rendimiento mucho mejor en términos de precisión en las predicciones en comparación con el método Chain-Ladder. Un MAPE del 0.93% es bastante bajo y sugiere que el modelo final está haciendo predicciones muy cercanas a los valores reales en el conjunto de datos evaluado.

- iii) Conclusión acerca de los objetivos comerciales se han alcanzado.

Después de realizar el modelo chain ladder, proponer regresiones lineales, Lasso y Ridge y luego de aplicar la validación cruzada, se concluye que la regresión de Lasso brinda una mejor aproximación al modelo que el método Chain Ladder.

Es importante tener claridad que en la industria trabajan con Chain Ladder, pero después de hacer el proceso se puede sugerir realizar estos modelos de predicción utilizando la regresión de Lasso. Los resultados presentados anteriormente sugieren la implementación de esta regresión, aunque sería muy

importante realizar una prueba de este método sobre otro conjunto de datos que no estén tan depurados ni estudiados.

También, es claro que se debe explorar otro método que brinde mejores resultados. La importancia de este modelo consiste en que preliminarmente da un mejor dato sobre la reserva que se debe tener para, por un lado poder responder por todas las reclamaciones que puedan existir en el sistema, y por otro lado, tener la oportunidad de trabajar el dinero sobrante en otros negocios o proyectos. El grupo de trabajo logró implementar un proyecto novedoso que permite abrir la exploración e implementación de este método. Sin embargo, no se debe pensar que es el mejor método que existe. Un trabajo a futuro podría ser realizar la implementación utilizando redes neuronales, observar el rendimiento y comparar con los resultados obtenidos y así tener o un respaldo con la regresión Lasso, o un nuevo modelo óptimo con las redes neuronales. También se pueden hacer mejoras en la codificación para optimizar el tiempo de ejecución o, quizá, hacerlo más simple.

Se espera que el proyecto pueda ser implementado en al menos una aseguradora y comprobar los resultados. Se considera que, a pesar que la codificación tomó un conjunto de datos para entrenamiento y otro para evaluación, se entiende que los datos estaban bien estructurados. Por eso el afán de usar otro tipo de datos.

También, queda la incertidumbre de observar si sirve para otro tipo de seguros y no solo los de las reclamaciones por negligencias médicas: Seguros vehiculares, seguros laborales, entre otros.