## **Segundo modelo**

Estos resultados de arriba son buenos, pero se pueden mejorar.

Transformaciones sobre las variables que pueden ser de utilidad.

df\_second = df.copy()  
df\_second['age2'] = df\_second.age\*\*2  
df\_second['sobrepeso'] = (df\_second.bmi >= 30).astype(int)  
df\_second['sobrepeso\*fumador'] = df\_second.sobrepeso \* df\_second.smoker\_yes

SI TIENE BMI > 30 SE PONE 1 SINO 0

Analizando el segundo modelo

X\_cols = ['sobrepeso\*fumador', 'smoker\_yes', 'age2', 'children']  
y\_col = ['charges']  
  
X = df\_second[X\_cols].values  
y = df\_second[y\_col].values  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y)  
sc\_x = StandardScaler().fit(X)  
sc\_y = StandardScaler().fit(y)  
  
X\_train = sc\_x.transform(X\_train)  
X\_test = sc\_x.transform(X\_test)

y\_train = sc\_y.transform(y\_train)  
y\_test = sc\_y.transform(y\_test)  
  
model = LinearRegression(fit\_intercept=False)  
model.fit(X\_train,y\_train)  
y\_pred = model.predict(X\_test)

mse = metrics.mean\_squared\_error(y\_test,y\_pred)  
r2 = metrics.r2\_score(y\_test, y\_pred)  
  
print("r2 ", r2.round(4))  
print("mse: ", mse.round(4))

r2 0.8319

mse: 0.1763

model.coef\_ = model.coef\_.reshape(-1)

y\_test = y\_test.reshape(-1)  
  
print("==========Summary==========")  
stats.summary(model, X\_test, y\_test, X\_cols)

Table

Description automatically generated

residuals = np.subtract(y\_test, y\_pred.reshape(-1))

plt.scatter(y\_pred, residuals)

plt.show()

**Chart, scatter chart

Description automatically generated**