probit

November 6, 2025

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
from fredapi import Fred
import yfinance as yf
from dotenv import load_dotenv
import os
```

[2]:
```
fred =Fred(os.getenv("fred_api"))
```

[3]:
```
def series_fred(fred, series_dict):
    data= pd.DataFrame()
    for code, name in series_dict.items():
        data[name] = fred.get_series(code, frequency = 'm')
    return data
```

[4]:
```
series={'SAHMCURRENT':'sahm_rule',
        'VIXCLS':'vix',
        'T10Y2Y':'T10Y2Y',
        'T10Y3M':'T10Y3M',
        'INDPRO':'INDPRO',
        'CPIAUCSL':'CPI',
        'PCE':'PCE'
        }
```

[5]:
```
sp_500 = yf.download(tickers='^GSPC', start='1990-01-01', end='2025-10-05',␣
 ↪interval='1mo', auto_adjust=True)['Close']
sp_500 = sp_500.round(4)
```

```
[**********************100%**********************]  1 of 1 completed
```

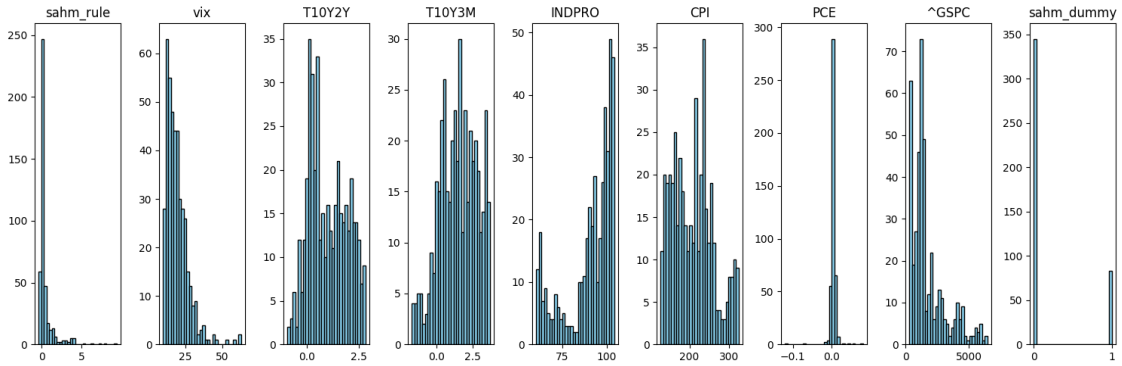
[6]:
```
data = series_fred(fred, series)
data = pd.merge(data, sp_500, left_index=True, right_index=True, how='left')
data['sahm_dummy'] = (data['sahm_rule'] > 0.5).astype(int)
data['PCE'] = np.log(data['PCE']/data['PCE'].shift(1))
data = data.dropna(axis=0)
```

```
data
```

[6]:
```
            sahm_rule    vix  T10Y2Y  T10Y3M    INDPRO      CPI       PCE  \
1990-01-01       0.27  23.35    0.12    0.31   61.6352  127.500  0.012542
1990-02-01       0.23  23.26    0.10    0.47   62.1951  128.000 -0.000670
1990-03-01       0.17  20.06   -0.04    0.42   62.4916  128.600  0.007136
1990-04-01       0.17  21.40    0.06    0.75   62.3511  128.900  0.004013
1990-05-01       0.20  18.10    0.12    0.75   62.5353  129.100  0.001537
…                 …     …       …       …        …        …        …
2025-04-01       0.27  31.97    0.50   -0.04  103.6224  320.321  0.003061
2025-05-01       0.27  20.46    0.50    0.06  103.6570  320.580  0.000414
2025-06-01       0.17  18.40    0.49   -0.04  104.2115  321.500  0.005449
2025-07-01       0.10  16.38    0.51   -0.02  103.8194  322.132  0.005462
2025-08-01       0.13  15.75    0.56   -0.04  103.9203  323.364  0.006139

                ^GSPC  sahm_dummy
1990-01-01   329.0800           0
1990-02-01   331.8900           0
1990-03-01   339.9400           0
1990-04-01   330.8000           0
1990-05-01   361.2300           0
…               …          …
2025-04-01  5569.0601           0
2025-05-01  5911.6899           0
2025-06-01  6204.9502           0
2025-07-01  6339.3901           0
2025-08-01  6460.2598           0

[428 rows x 9 columns]
```

[7]:
```
variables = data.columns
n =  len(variables)
```

[8]:
```
fig, axes = plt.subplots(1, n, figsize=(15,5))

for i, col in enumerate(variables):
    axes[i].hist(data[col], bins=30, color="skyblue", edgecolor="black")
    axes[i].set_title(f"{col}")

plt.tight_layout()
plt.show()
```



[9]:
```
#Modelo Probit
x = data.iloc[:,1:-1]
x = sm.add_constant(x)
y = data.iloc[:,-1]
probit_model = sm.Probit(y, x)
res = probit_model.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.288255
         Iterations 7
```

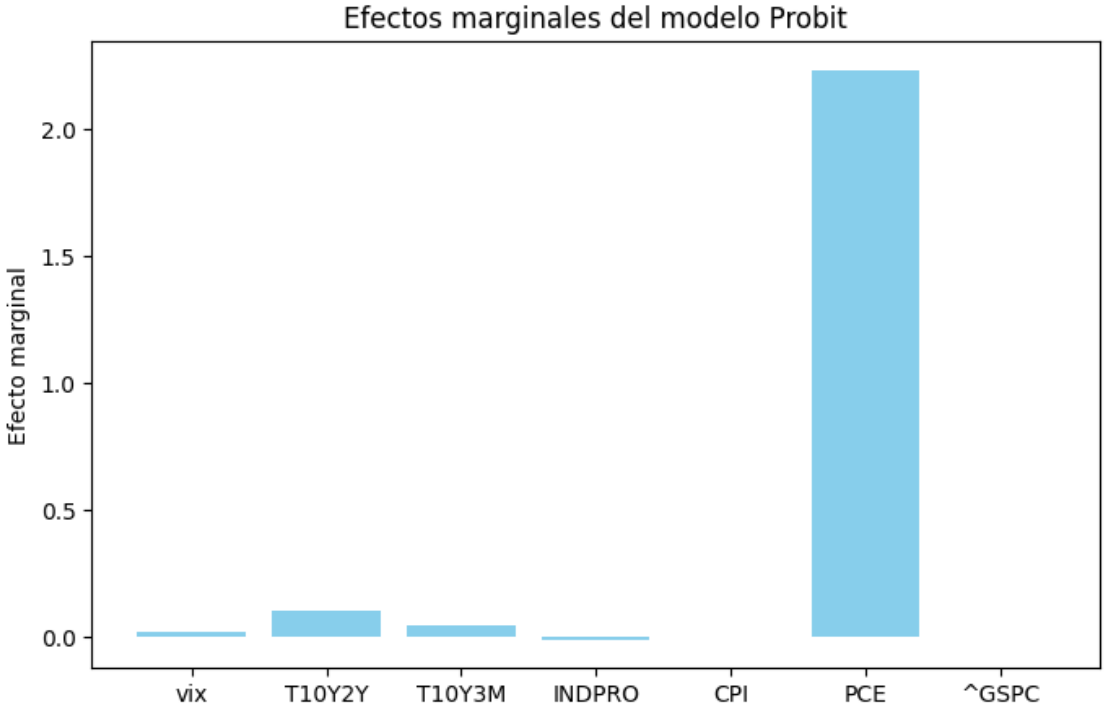
[18]:
```
print(res.summary())
```

```
                          Probit Regression Results
==============================================================================
Dep. Variable:             sahm_dummy   No. Observations:                  428
Model:                         Probit   Df Residuals:                      420
Method:                           MLE   Df Model:                            7
Date:                mar., 04 nov. 2025   Pseudo R-squ.:                  0.4140
Time:                        12:02:02   Log-Likelihood:                -123.37
converged:                       True   LL-Null:                       -210.52
Covariance Type:            nonrobust   LLR p-value:                 3.126e-34
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2823      0.767      0.368      0.713      -1.221       1.785
vix            0.1136      0.015      7.475      0.000       0.084       0.143
T10Y2Y         0.6235      0.278      2.245      0.025       0.079       1.168
T10Y3M         0.2602      0.210      1.238      0.216      -0.152       0.672
INDPRO        -0.0724      0.014     -5.008      0.000      -0.101      -0.044
CPI            0.0035      0.008      0.460      0.646      -0.011       0.018
PCE           13.8463      7.429      1.864      0.062      -0.713      28.406
^GSPC          0.0005      0.000      2.088      0.037    2.81e-05       0.001
==============================================================================
```

[19]:
```
margeff = res.get_margeff()
print(margeff.summary())
```

```
       Probit Marginal Effects
=====================================
Dep. Variable:             sahm_dummy
Method:                           dydx
At:                           overall
================================================================================
                dy/dx    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
vix             0.0183      0.002      9.645      0.000       0.015       0.022
T10Y2Y          0.1005      0.044      2.304      0.021       0.015       0.186
T10Y3M          0.0420      0.034      1.238      0.216      -0.024       0.108
INDPRO         -0.0117      0.002     -5.547      0.000      -0.016      -0.008
CPI             0.0006      0.001      0.461      0.645      -0.002       0.003
PCE             2.2328      1.176      1.899      0.058      -0.072       4.538
^GSPC        7.39e-05   3.51e-05      2.107      0.035    5.17e-06       0.000
================================================================================
```

[20]:
```
effects = margeff.margeff
variables = x.columns.drop('const')
plt.figure(figsize=(8,5))
plt.bar(variables, effects, color='skyblue')
plt.ylabel("Efecto marginal")
plt.title("Efectos marginales del modelo Probit")
plt.show()
```



[13]:
```
#estadistica descriptiba

describe_stats = data.describe().T.round(4)
```

[14]:
```
#modelo logit

x_logit = data.iloc[:,1:-1]
x_logit = sm.add_constant(x)
y_logit = data.iloc[:,-1]
```

[15]:
```
logit_model = sm.Logit(y,x)
res_logit = logit_model.fit()
print(res_logit.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.287247
         Iterations 8
                           Logit Regression Results
==============================================================================
Dep. Variable:             sahm_dummy   No. Observations:                  428
Model:                          Logit   Df Residuals:                      420
Method:                           MLE   Df Model:                            7
Date:                Tue, 04 Nov 2025   Pseudo R-squ.:                  0.4160
Time:                        12:01:39   Log-Likelihood:                -122.94
converged:                       True   LL-Null:                       -210.52
Covariance Type:            nonrobust   LLR p-value:                 2.055e-34
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.6222      1.372      0.453      0.650      -2.067       3.312
vix            0.2067      0.029      7.072      0.000       0.149       0.264
T10Y2Y         0.9836      0.494      1.993      0.046       0.016       1.951
T10Y3M         0.6037      0.385      1.569      0.117      -0.150       1.358
INDPRO        -0.1299      0.027     -4.896      0.000      -0.182      -0.078
CPI            0.0045      0.013      0.331      0.741      -0.022       0.031
PCE           22.1281     12.974      1.706      0.088      -3.300      47.556
^GSPC          0.0009      0.000      2.242      0.025       0.000       0.002
==============================================================================
```

[16]:
```
#Resultados log_odds

odds_ratio_logit = np.exp(res_logit.params)
print(odds_ratio_logit)
```

```
const     1.863051e+00
vix       1.229631e+00
```

```
T10Y2Y    2.673956e+00
T10Y3M    1.828792e+00
INDPRO    8.782102e-01
CPI       1.004472e+00
PCE       4.074992e+09
^GSPC     1.000886e+00
dtype: float64
```

[17]:
```
#Efectos marginales logit

marginal_logit = res_logit.get_margeff()
print(marginal_logit.summary())
```

```
        Logit Marginal Effects
=====================================
Dep. Variable:             sahm_dummy
Method:                          dydx
At:                           overall
================================================================================
                dy/dx    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
vix            0.0185      0.002      9.679      0.000       0.015       0.022
T10Y2Y         0.0878      0.043      2.034      0.042       0.003       0.172
T10Y3M         0.0539      0.034      1.579      0.114      -0.013       0.121
INDPRO        -0.0116      0.002     -5.606      0.000      -0.016      -0.008
CPI            0.0004      0.001      0.331      0.740      -0.002       0.003
PCE            1.9762      1.140      1.733      0.083      -0.259       4.211
^GSPC       7.911e-05   3.49e-05      2.270      0.023    1.08e-05       0.000
================================================================================
```

[ ]: