**Regionalization of annual precipitation applying support vector regression**

Efrain Noa-Yarasca

**Regionalización de la precipitación anual aplicando regresión de vectores de soporte**

Efrain Noa-Yarasca

1. **Introducción**

En el presente trabajo se evalúa la relación que existe entre la altitude y la precipitación annual de la Cuenca del río Cachi aplicando una técnica clasica de regression simple y la Regression de Vectores de Soporte (RVS) que es una técnica de Machine Learning (ML)

**Estudio de caso**

La cuenca del río Cachi presenta doce estaciones de precipitación, las cuales se encuentran espacialmente distribuidas.

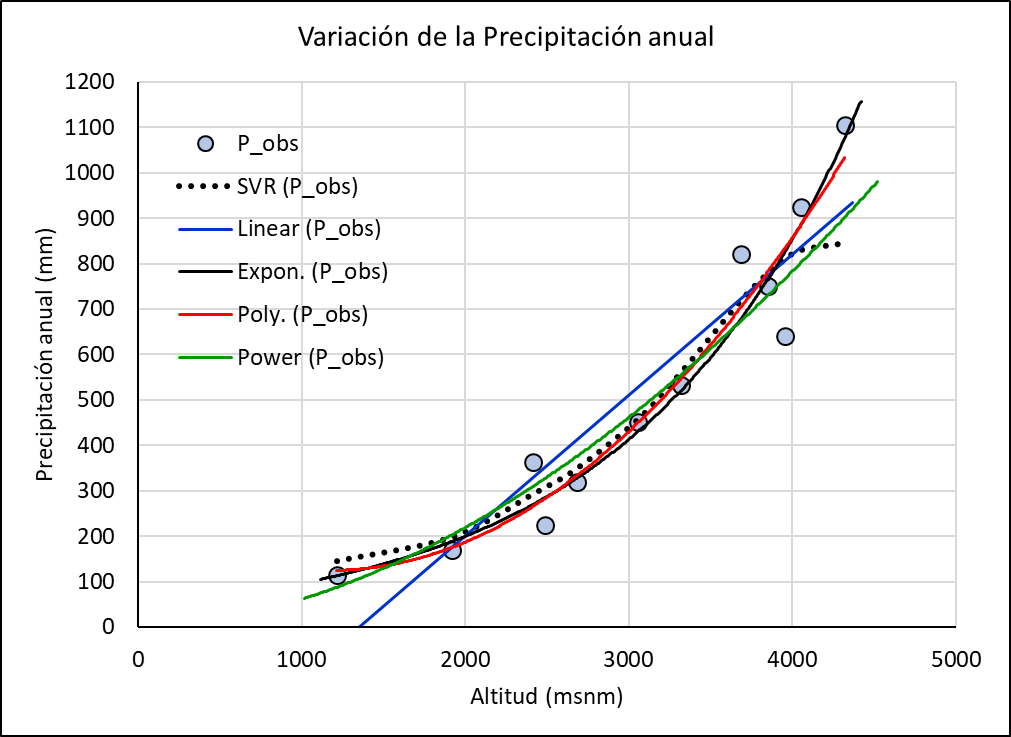
1. **Metodología**

El modelo de RVS fue implementado en el programa python empleando la plataforma cloud de Google Colab. El codigo del programa puede encontrarse en la siguiente dirección. <https://github.com/noayarae/Machine_Learning_notes.git>

1. **Resultados**

La altitude de las estaciones variaron de 1215 a 4320 msnm, la precipitación annual variá de 115.2 a 1105.1 mm.





Modelado

Las ecuaciones de ajuste fueron obtenidos empleando la hoja cálculo Excel

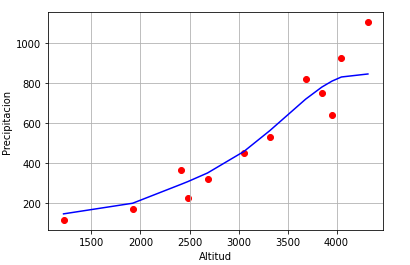
Modelo lineal

Modelo exponencial

Modelo polynomial

Modelo de potencia

El modelo SVR provee el resultado mostrado en la Figura 2



Los resultados de la predicción se muestran en ls siguiente table



Los valores de la Raiz del Error Cuadratico Medio se muestran en la siguiente table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Métrica | RL | RE | RPl | RPt | SVR |
| RMSE | 108.9 | 97.1 | 81.3 | 91.6 | 104.6 |
| R | 0.933 | 0.968 | 0.967 | 0.958 | 0.946 |

1. **Conclusiones**

El modelo de Regresion de soporte vectorial prove mejora rendimiento que el modelo linear pero menor a los otros modelos (RE, RPl, RPt).

1. **Referencias**

Codigo

*# Importing the libraries*

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

**import** **pandas** **as** **pd**

*# Importing the dataset WITH transf*

dataset = pd.read\_csv('precip.csv')

a = dataset.iloc[:, 2:3].values

b = dataset.iloc[:, 3].values

*#print (a), print (b)*

*# Feature Scaling*

**from** **sklearn.preprocessing** **import** StandardScaler

sc\_a = StandardScaler()

sc\_b = StandardScaler()

a2 = sc\_a.fit\_transform(a)

*#y = sc\_b.fit\_transform(b)*

b2 = sc\_b.fit\_transform(b.reshape(-1, 1))

pv = 3520

pvt1 = (pv-a.mean())/a.std()

print (pvt1)

*#pvt2 = sc\_a.fit\_transform(a)*

*# Fitting SVR to the dataset (Old, New)*

**from** **sklearn.svm** **import** SVR

regressor = SVR(kernel = 'rbf')

regressor.fit(a2, b2)

*# Predicting a new result (Old)*

print ("pred1")

y\_pred = regressor.predict(a2)

print (y\_pred)

y\_pred = sc\_b.inverse\_transform(y\_pred)

print (y\_pred)

print ("pred2")

y\_pred1 = regressor.predict([[pvt1]]) *#y\_pred1 = regressor.predict([[0.348]])*

print (y\_pred1),

y\_pred1 = sc\_b.inverse\_transform(y\_pred1)

print (y\_pred1)

*# Visualising the SVR results (New)*

plt.scatter(X, y, color = 'red')

*#sc\_b.inverse\_transform()*

plt.plot(X, y\_pred, color = 'blue')

plt.xlabel('Altitud')

plt.ylabel('Precipitacion')

plt.grid()

*#grid(color='r', linestyle='-', linewidth=2)*

plt.show()