Variabilidad en el ciclo anual de la velocidad de flujo en una zona del mar caribe

Paula Andrea Espinosa Ordoñez

Agenda

- Conceptualización
- Aplicación de métodos
- Resultados
- Conclusiones

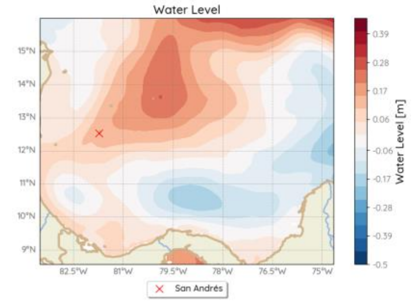
PROBLEMA

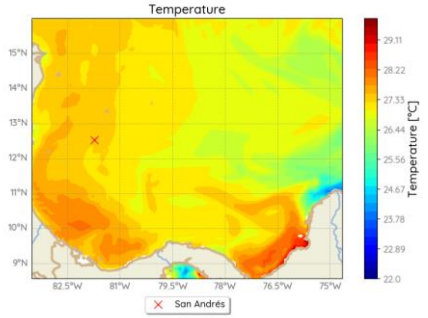
Variables Independientes (X):

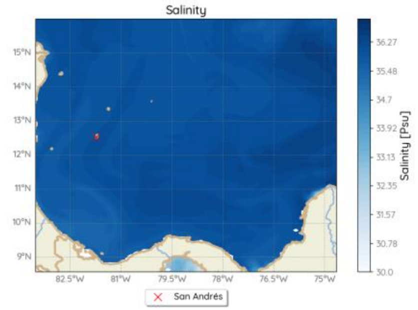
- Nivel del agua
- Salinidad
- Temperatura
- Altura y dirección de ola
- Magnitud y dirección del viento

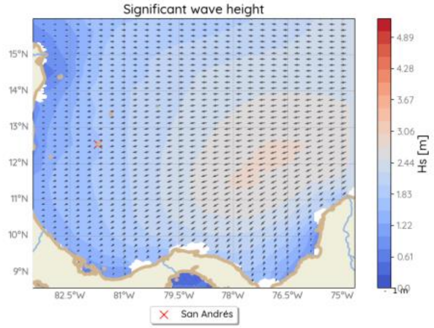
Variables Dependiente (Y):

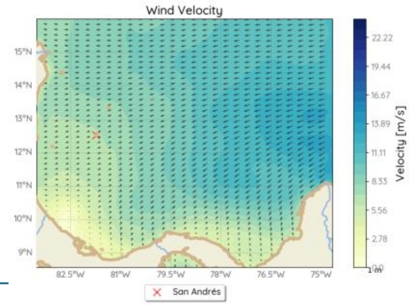
- Magnitud de la velocidad
- Dirección de la velocidad







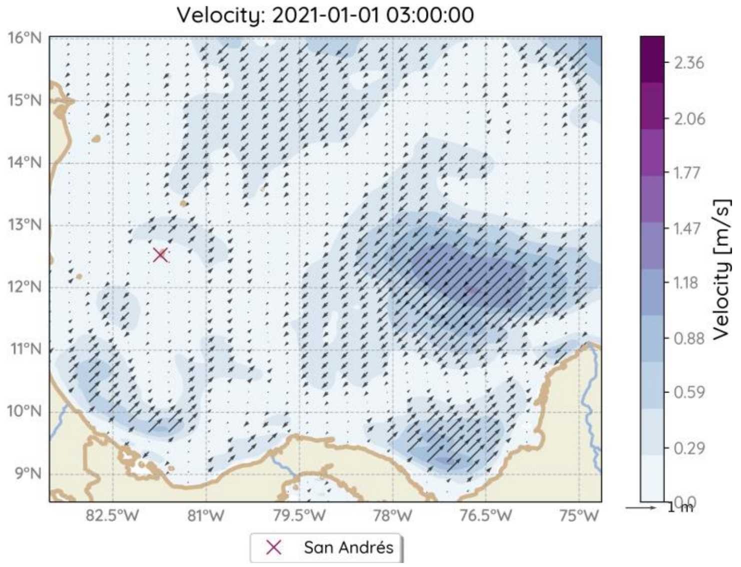




Resolución:

- Pixeles: 4km x 8 km
- Temporal: promedio mensual
- Hycom
- ERA5

Predicciones estacionales de Velocidad de flujo



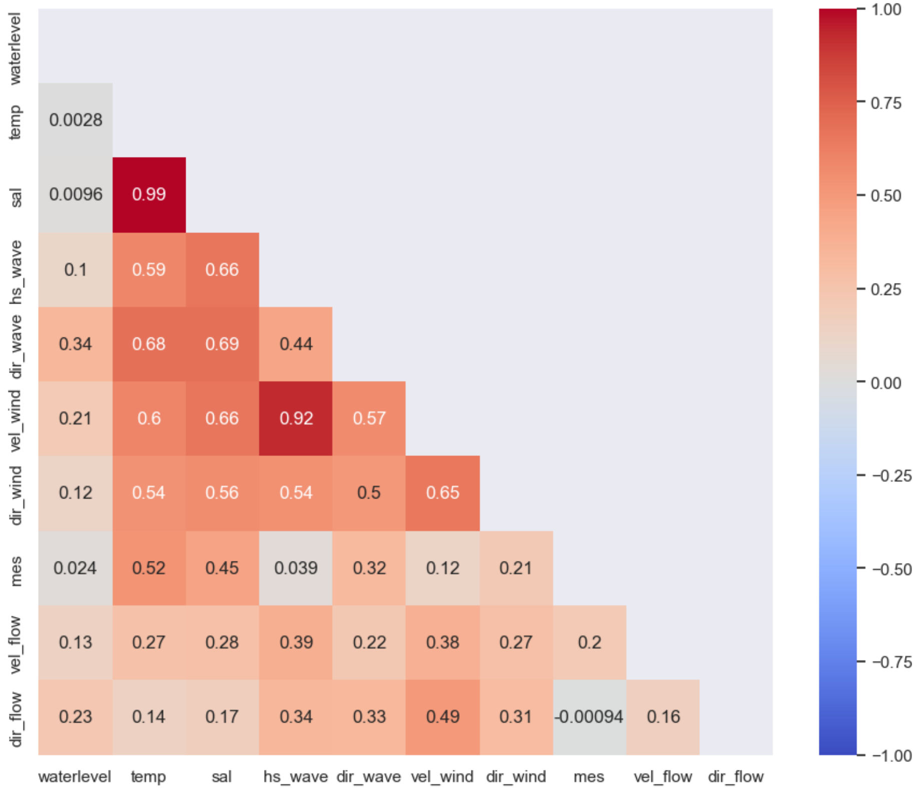
UNIVERSIDAD NACIONAL DE COLOMBIA

Datos

	index	waterlevel	temp	sal	hs_wave	dir_wave	vel_wind	dir_wind	u_wind	v_wind	vel_flow
0	46	0.260220	0.260220	0.260220	0.116837	0.116837	0.116837	45.000000	0.116837	0.116837	0.368006
1	47	0.263163	0.263163	0.263163	0.122465	0.122465	0.122465	45.000000	0.122465	0.122465	0.372168
2	48	0.258976	0.258976	0.258976	0.130494	0.130494	0.130494	45.000000	0.130494	0.130494	0.366247
3	49	0.256825	0.256825	0.256825	0.140012	0.140012	0.140012	45.000000	0.140012	0.140012	0.363206
4	50	0.258829	0.258829	0.258829	0.149988	0.149988	0.149988	45.000000	0.149988	0.149988	0.366040
...	...	...	...	...	...	...	...	...			
221371	20939	0.218121	28.110162	35.755665	1.887047	87.019027	9.213161	259			
221372	20940	0.222498	28.105264	35.748895	1.888027	87.102830	9.229263	259			
221373	20941	0.226166	28.097134	35.737414	1.888999	87.187034	9.245683	259			
221374	20942	0.229069	28.081057	35.720507	1.889970	87.271393	9.257756	259			
221375	20943	0.231113	28.077571	35.710580	1.890939	87.355596	9.262286	259			

221376 rows × 15 columns

dir_flow	u_flow	v_flow	mes
45.0	0.260220	0.260220	1
45.0	0.263163	0.263163	1
45.0	0.258976	0.258976	1
45.0	0.256825	0.256825	1
45.0	0.258829	0.258829	1
...	...	...	...
225.0	-0.552490	-0.552490	12
225.0	-0.567478	-0.567478	12
225.0	-0.581518	-0.581518	12
225.0	-0.592680	-0.592680	12
225.0	-0.599862	-0.599862	12



1

SELECCIÓN DE VARIABLES

A través del método **Recursive Feature Elimination (RFE)**

UNIVERSIDAD NACIONAL DE COLOMBIA

Aplicación de métodos- Esquema

1 SELECCIÓN DE VARIABLES

A través del método **Recursive Feature Elimination (RFE)**

2 K-FOLD

Evaluar el rendimiento general del modelo sin entrar en detalle.

3 HIPERPARÁMETROS

Evaluar el selección de hiperparámetros dependiendo del modelo (paramétrico o no paramétrico)

4 APLICACIÓN DEL MÉTODO

Evaluar el modelo con los datos de entrenamiento y **testeo**.

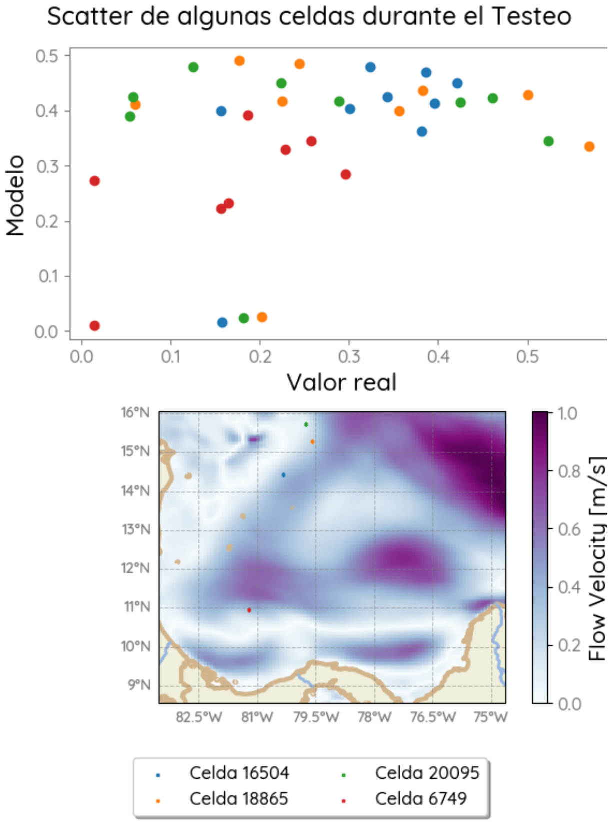
Aplicación de métodos – Regresión Lineal

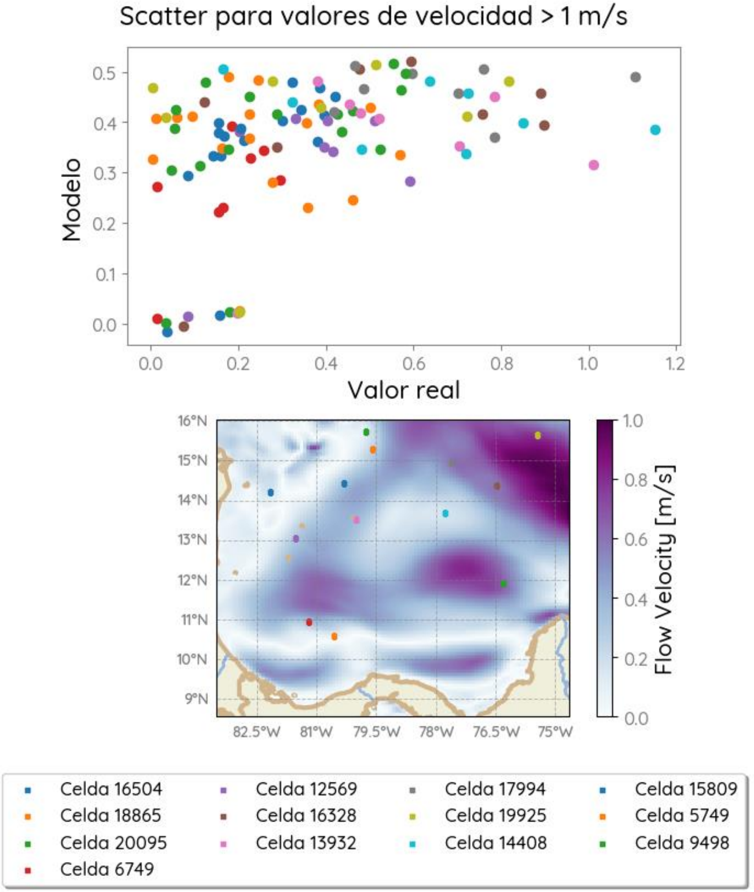
ENTRENAMIENTO

```
                            OLS Regression Results
=======================================================================================
Dep. Variable:               vel_flow   R-squared (uncentered):                   0.701
Model:                            OLS   Adj. R-squared (uncentered):              0.701
Method:                 Least Squares   F-statistic:                          1.014e+05
Date:                Tue, 06 Dec 2022   Prob (F-statistic):                        0.00
Time:                        13:33:14   Log-Likelihood:                         -111.29
No. Observations:              173085   AIC:                                      230.6
Df Residuals:                  173081   BIC:                                      270.8
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
waterlevel     0.1911      0.005     41.805      0.000       0.182       0.200
temp           0.0049   6.97e-05     70.239      0.000       0.005       0.005
hs_wave        0.1408      0.003     51.793      0.000       0.135       0.146
vel_wind       0.0046      0.001      7.831      0.000       0.003       0.006
==============================================================================
Omnibus:                    11274.961   Durbin-Watson:                   2.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            13624.035
Skew:                           0.664   Prob(JB):                         0.00
Kurtosis:                       3.357   Cond. No.                         222.
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

R2
0.151

VALIDACIÓN





Sin incluir el intercepto, Problemas de alto bias

Aplicación de métodos - Regresión Lineal

Incluyendo el intercepto

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               vel_flow   R-squared:                       0.164
Model:                            OLS   Adj. R-squared:                  0.164
Method:                 Least Squares   F-statistic:                     8475.
Date:                Tue, 06 Dec 2022   Prob (F-statistic):               0.00
Time:                        15:30:12   Log-Likelihood:                 637.40
No. Observations:              173085   AIC:                            -1265.
Df Residuals:                  173080   BIC:                            -1214.
Df Model:                           4
Covariance Type:            nonrobust
==================================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
----------------------------------------------------------------------------------
Intercept      0.0824      0.002     38.779      0.000       0.078       0.087
waterlevel     0.1837      0.005     40.305      0.000       0.175       0.193
temp           0.0022   9.91e-05     21.707      0.000       0.002       0.002
hs_wave        0.1346      0.003     49.670      0.000       0.129       0.140
vel_wind       0.0051      0.001      8.801      0.000       0.004       0.006
==================================================================================
Omnibus:                    13838.957   Durbin-Watson:                   2.005
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            17485.093
Skew:                           0.740   Prob(JB):                         0.00
Kurtosis:                       3.485   Cond. No.                         222.
==================================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

TEST:

```python
from sklearn.metrics import r2_score
y_pre_test1 = lm.predict(X_test.iloc[:,1:])
r2_test = r2_score(y_test1,y_pre_test1)
print('El r2 representando con los datos de validación es: ',r2_test)
```

✓ 0.4s

```
El r2 representando con los datos de validación es:  0.1685536886564295
```

Aplicación de métodos - Regresión Lineal

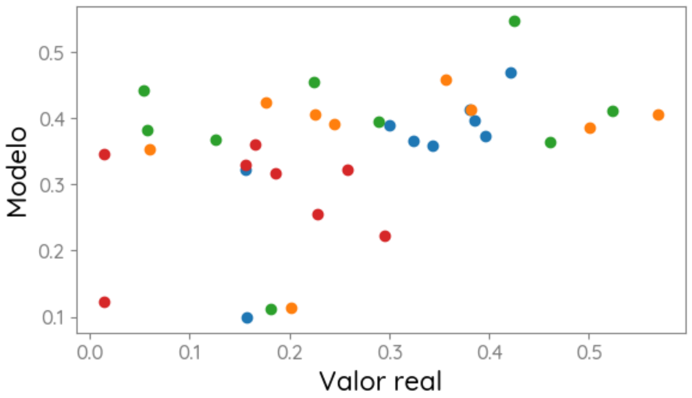
ENTRENAMIENTO

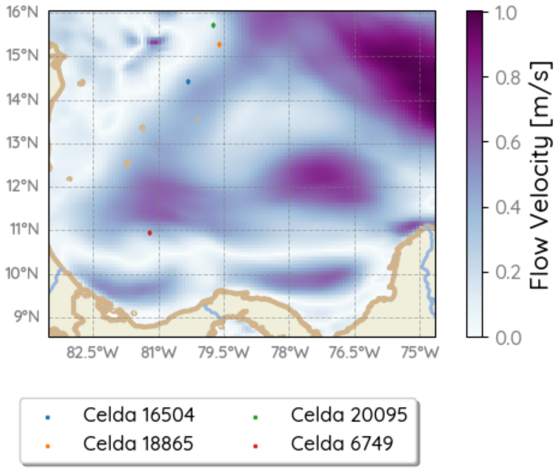
```
                            OLS Regression Results
==============================================================================
Dep. Variable:               vel_flow   R-squared:                       0.325
Model:                            OLS   Adj. R-squared:                  0.325
Method:                 Least Squares   F-statistic:                     5559.
Date:                Tue, 06 Dec 2022   Prob (F-statistic):               0.00
Time:                        16:21:25   Log-Likelihood:                 19191.
No. Observations:              173085   AIC:                        -3.835e+04
Df Residuals:                  173069   BIC:                        -3.819e+04
Df Model:                          15
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
waterlevel     0.2695      0.004     63.718      0.000       0.261       0.278
temp          -0.1125      0.002    -60.829      0.000      -0.116      -0.109
hs_wave        0.4674      0.003    141.349      0.000       0.461       0.474
vel_wind      -0.0361      0.001    -60.680      0.000      -0.037      -0.035
mes__1         0.0905      0.002     47.339      0.000       0.087       0.094
mes__2         2.7566      0.052     52.817      0.000       2.654       2.859
mes__3         2.8541      0.052     55.072      0.000       2.753       2.956
mes__4         2.9366      0.052     56.320      0.000       2.834       3.039
mes__5         3.0290      0.053     57.031      0.000       2.925       3.133
mes__6         3.0972      0.054     57.611      0.000       2.992       3.203
mes__7         3.0584      0.054     56.688      0.000       2.953       3.164
mes__8         3.3107      0.054     60.904      0.000       3.204       3.417
mes__9         3.3975      0.055     62.122      0.000       3.290       3.505
mes__10        3.3879      0.055     61.507      0.000       3.280       3.496
mes__11        3.2966      0.055     60.354      0.000       3.190       3.404
mes__12        2.9788      0.054     55.485      0.000       2.874       3.084
==============================================================================
Omnibus:                     6149.329   Durbin-Watson:                   2.004
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             7122.612
Skew:                           0.441   Prob(JB):                         0.00
Kurtosis:                       3.459   Cond. No.                     9.51e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 9.51e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

R2 0.31

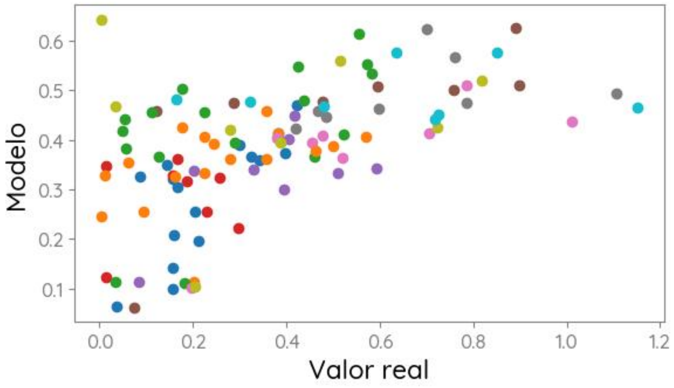
VALIDACIÓN

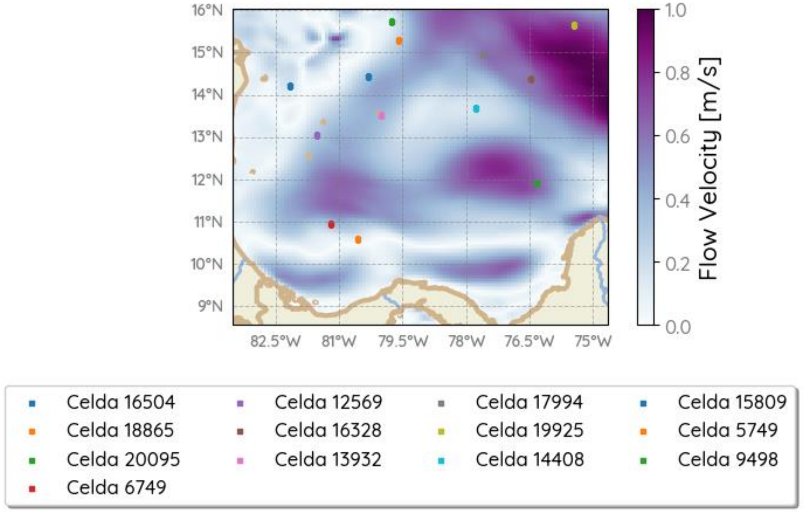
Scatter de algunas celdas durante el Testeo





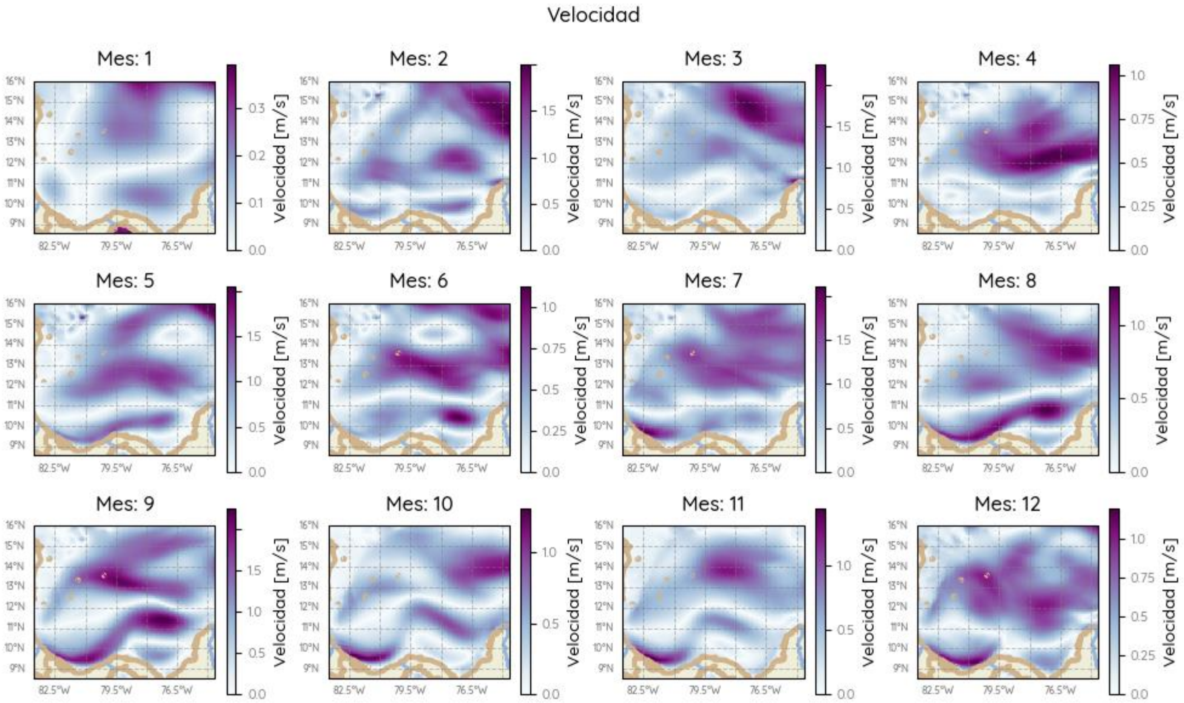
Scatter para valores de velocidad > 1 m/s

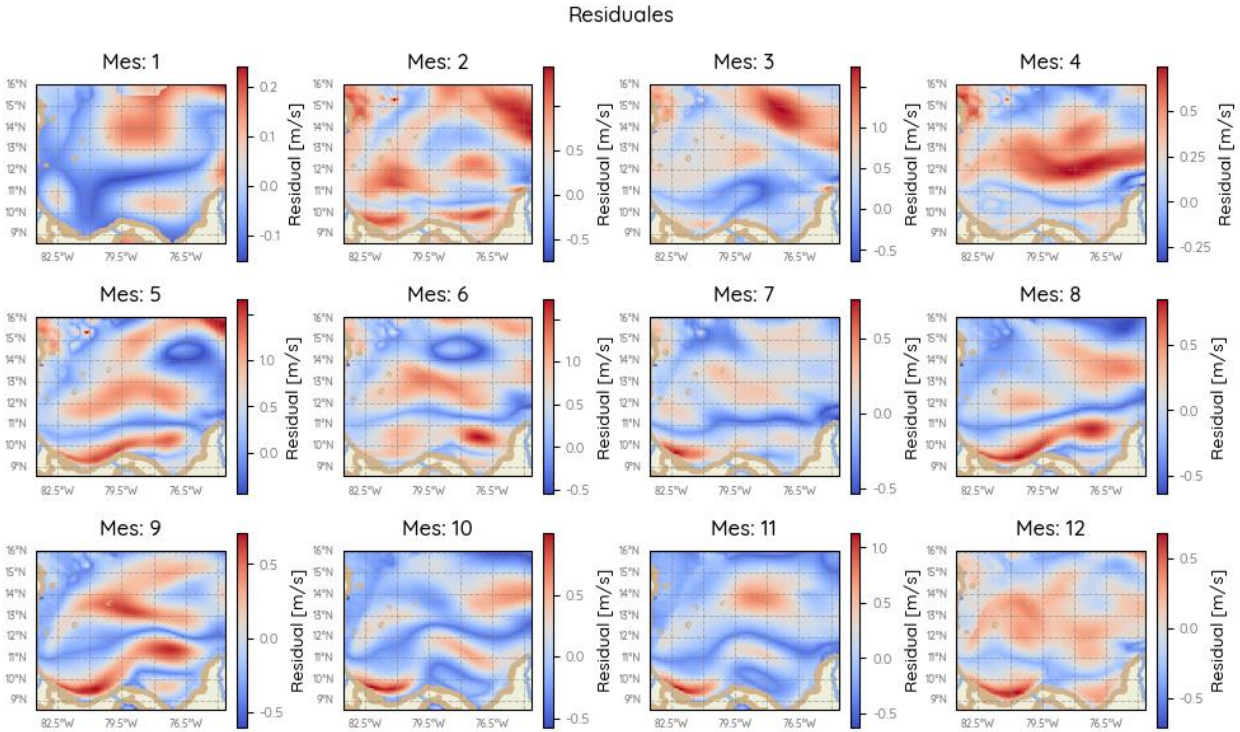




Aplicación de métodos - Regresión Lineal

EVALUACIÓN DEL LA REGRESIÓN LINEAL

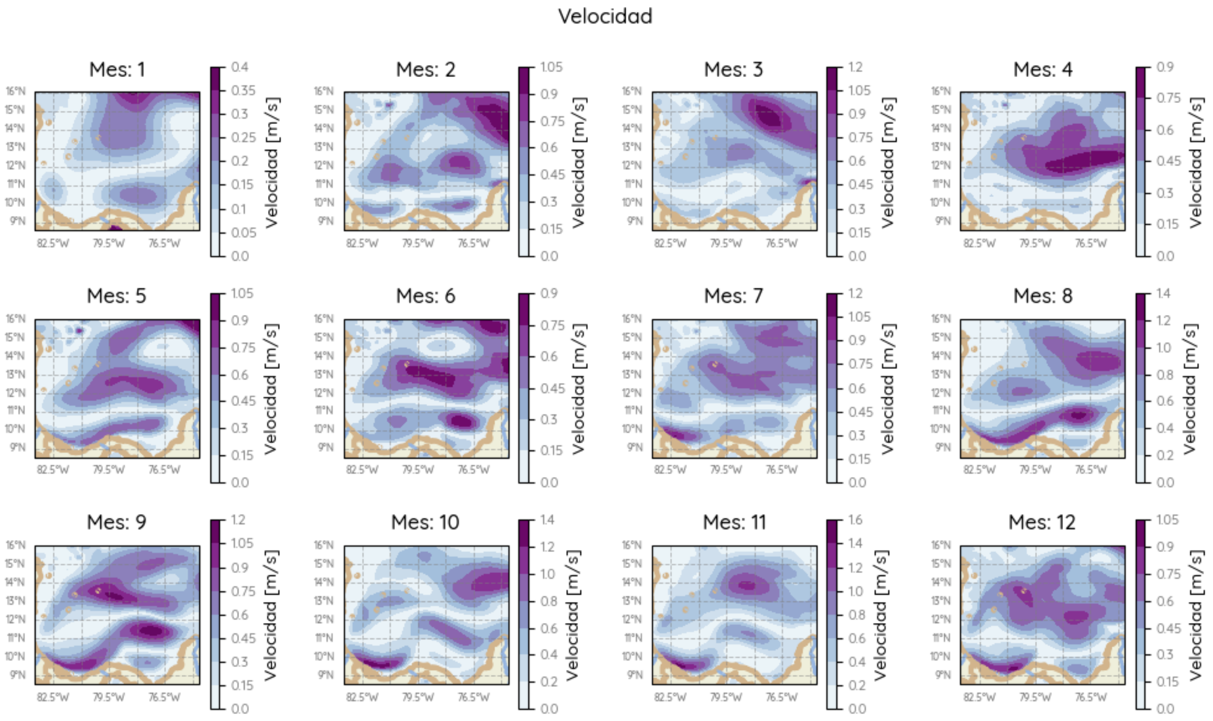




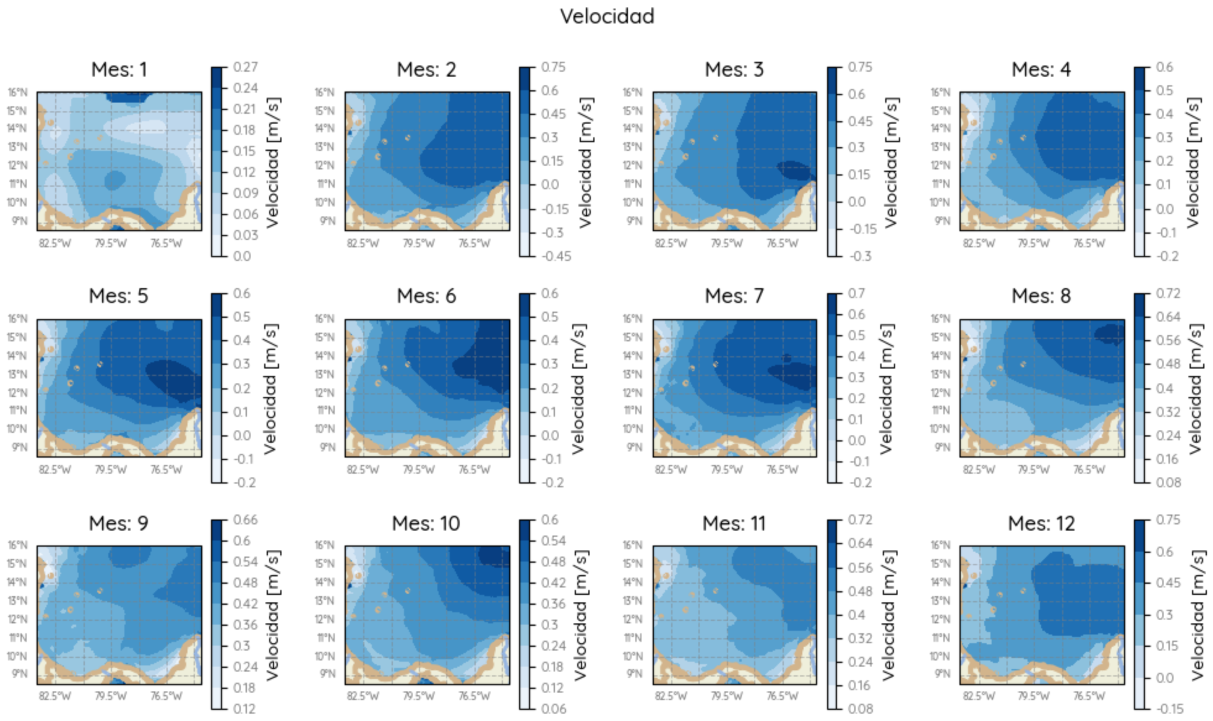
UNIVERSIDAD NACIONAL DE COLOMBIA

Aplicación de métodos - Regresión Lineal

Y : variable independiente

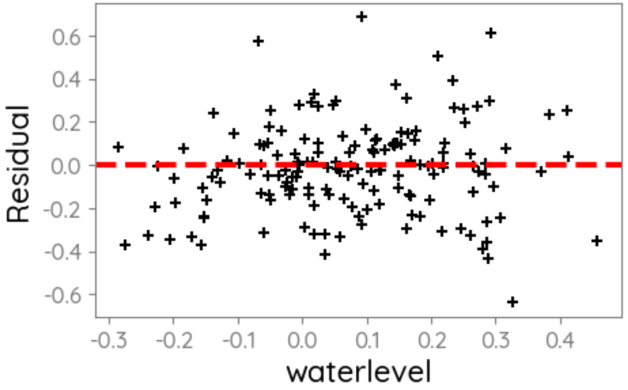


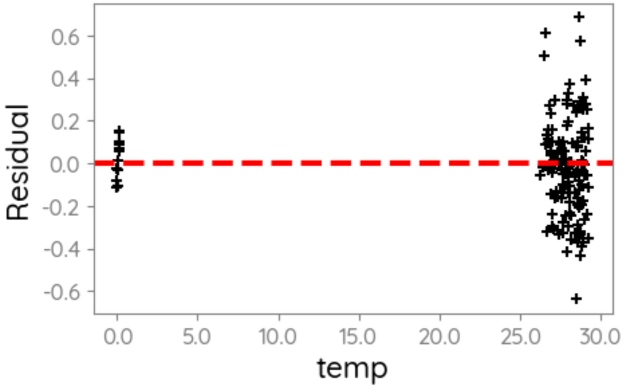
Predicción con OLS

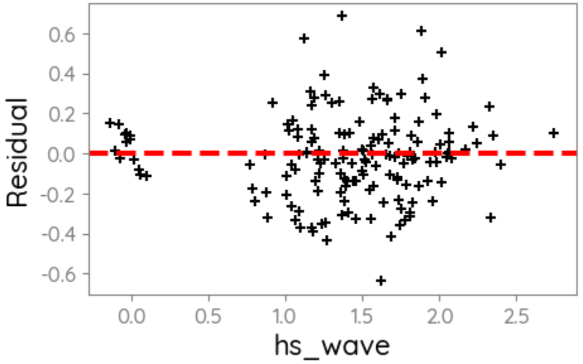


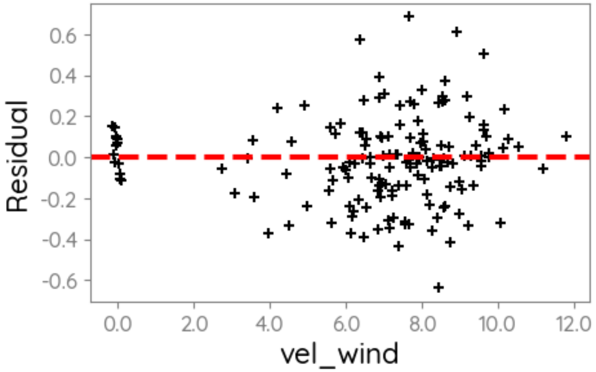
Aplicación de métodos - Regresión Lineal

EVALUACIÓN DEL LA REGRESIÓN LINEAL





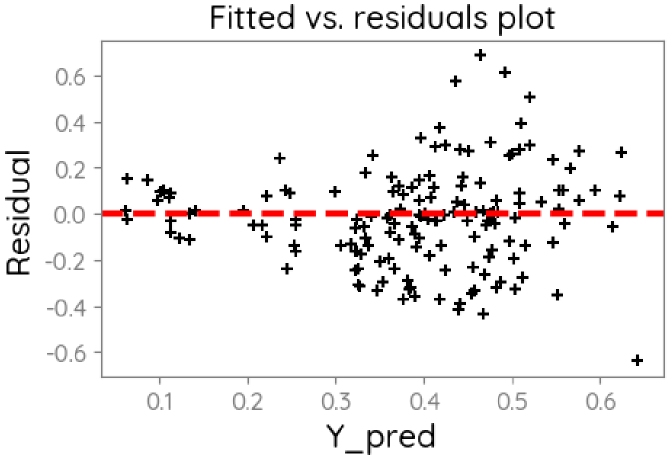


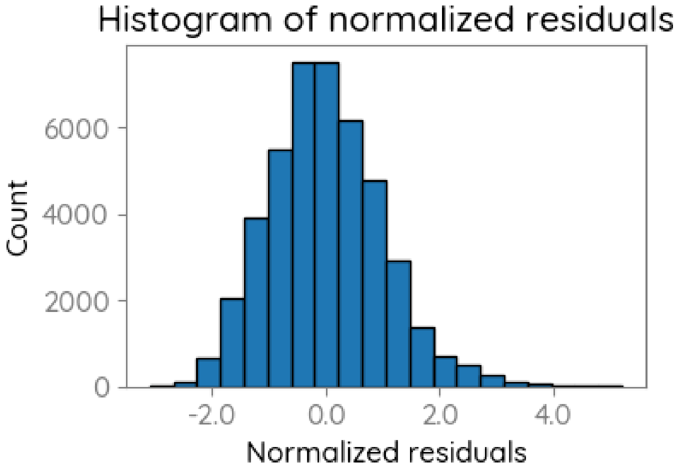


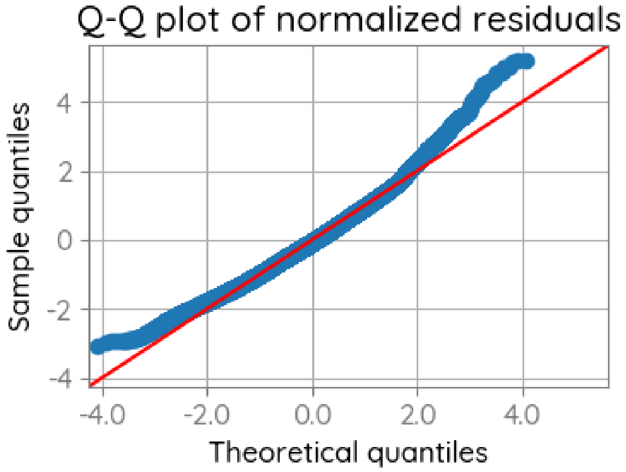
Consideraciones

Residuals will...

- have a constant variance
- be approximately normally distributed
- be independent of one another







Aplicación de métodos - Regresión Lineal

Aplicando Regularización

Lasso

```python
# RandomSearch
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import uniform

param_grid = {'alpha': uniform()}
rsearch = RandomizedSearchCV(estimator=Lasso(fit_intercept = False), param_distributions=param_grid, n_iter=10
rsearch.fit(X_train_sd.iloc[:,1:], y_train1)
print(rsearch.best_score_)
print(rsearch.best_estimator_.alpha)

```

✓ 10.8s Python

```
-1.5969334930207655
0.00011437481734488664
```

Ridge

```python
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import uniform

param_grid = {'alpha': uniform()}
rsearch = RandomizedSearchCV(estimator=Ridge(fit_intercept = False), param_distributions=param_grid, n_iter=10
rsearch.fit(X_train_sd.iloc[:,1:], y_train1)
print(rsearch.best_score_)
print(rsearch.best_estimator_.alpha)
```

✓ 7.9s Python

```
-1.596933141314826
0.9888610889064947
```

Aplicación de Métodos - KNN

Evaluación del desempeño del modelo

```python
# Validación curzada con shufflesplit
from sklearn.neighbors import KNeighborsRegressor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import ShuffleSplit

kfold = ShuffleSplit(n_splits=5)
model = KNeighborsRegressor()
results = cross_val_score(model, X_train.iloc[:,1:], y_train1, cv=kfold)
print(results.mean())
print(results.std())
```

✓ 1.5s

```
0.9129901863252652
0.0032970304178354636
```

2

K-FOLD

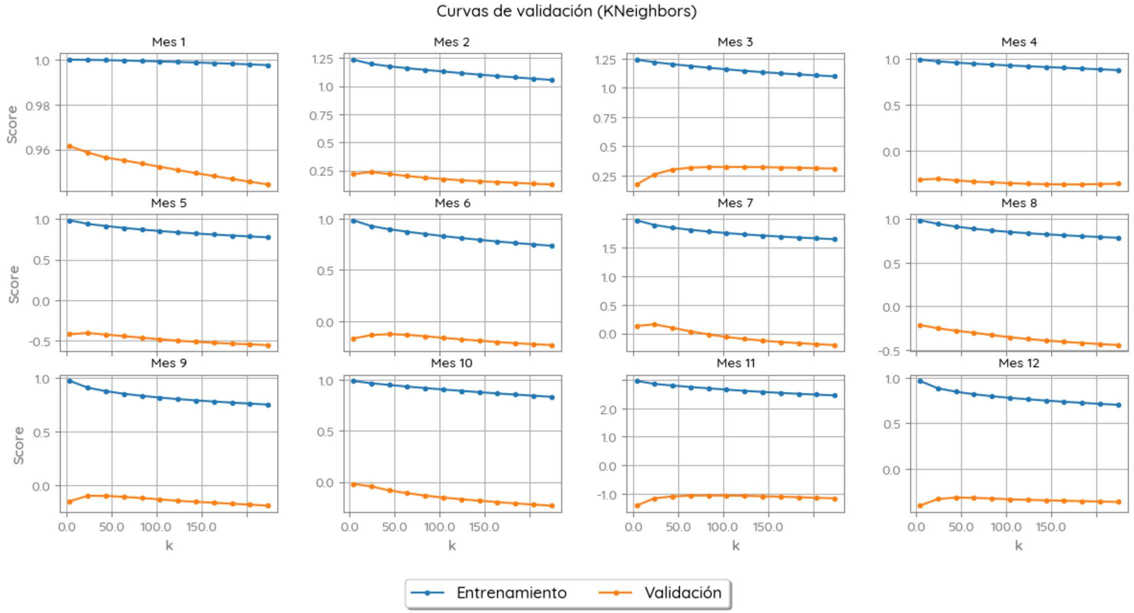
Evaluar el rendimiento general del modelo sin entrar en detalle.

Selección de hiperparámetro K neighbors

```python
# Buscada de los vecinos
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import uniform

k_neighbors = np.arange(2,115,10)
param_grid = {'n_neighbors': k_neighbors}
rsearch = RandomizedSearchCV(estimator=KNeighborsRegressor(), param_distributions=param_grid, n_iter=100, rando
rsearch.fit(X_train.iloc[:,1:], y_train1)
print('Mejor R2',rsearch.best_score_)
print('Mejor estimador', rsearch.best_estimator_.n_neighbors)
```

✓ 1m 0.6s Python

```
Mejor R2 0.9126437799970694
Mejor estimador 2
```



Aplicación de Métodos - KNN

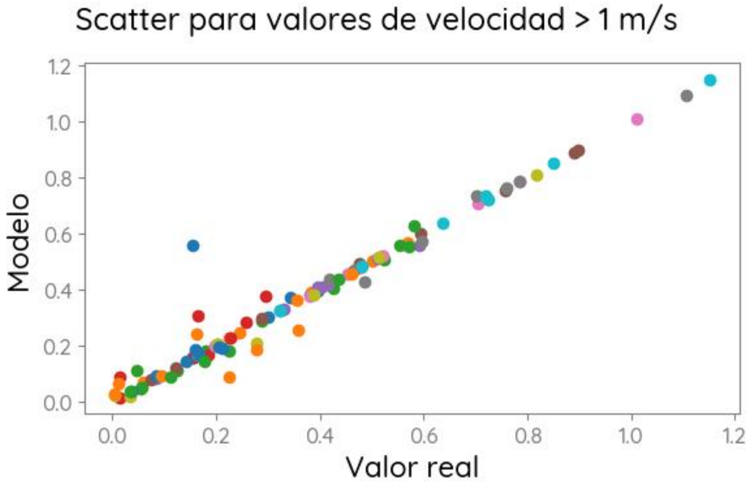
Proceso

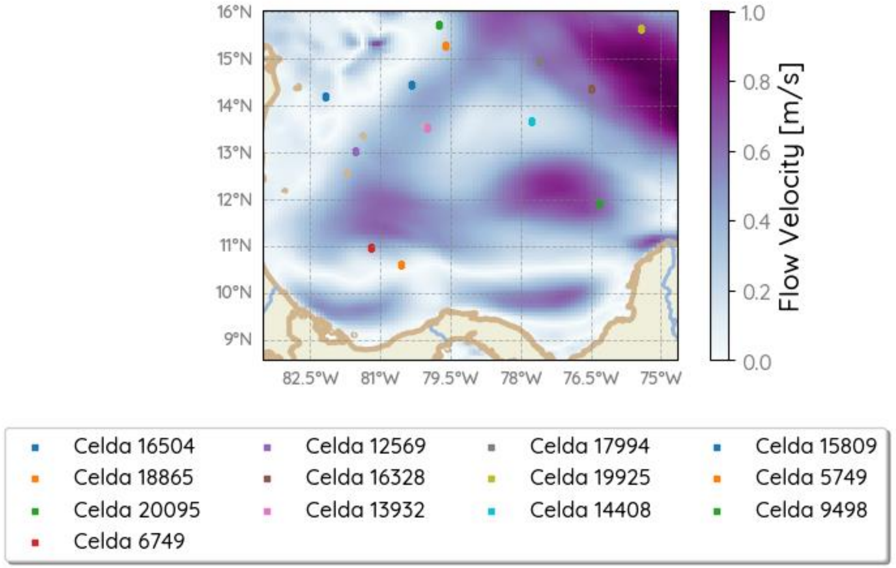
```python
import math
model = KNeighborsRegressor(n_neighbors=2)
# Función de ajuste
fun_ajuste = model.fit(X_train.iloc[:,1:],y_train1)

# Score del entrenamiento
score=model.score(X_train.iloc[:,1:],y_train1)
print('R2 del entrenamiento:',score)

# Score del test
y_pred_test_knn = model.predict(X_test.iloc[:,1:])
r2_test = r2_score(y_test1,y_pred_test_knn)
print('El R2 de la validación: ',r2_test)

mse =mean_squared_error(y_test1, y_pred_test_knn)
print("Mean Squared Error:",mse)
rmse = math.sqrt(mse)
print("Root Mean Squared Error:", rmse)
```

```
R2 del entrenamiento: 0.9767871276795833
El R2 de la validación:  0.9217751532563562
Mean Squared Error: 0.005472961625236615
Root Mean Squared Error: 0.073979467592276
```

Validación

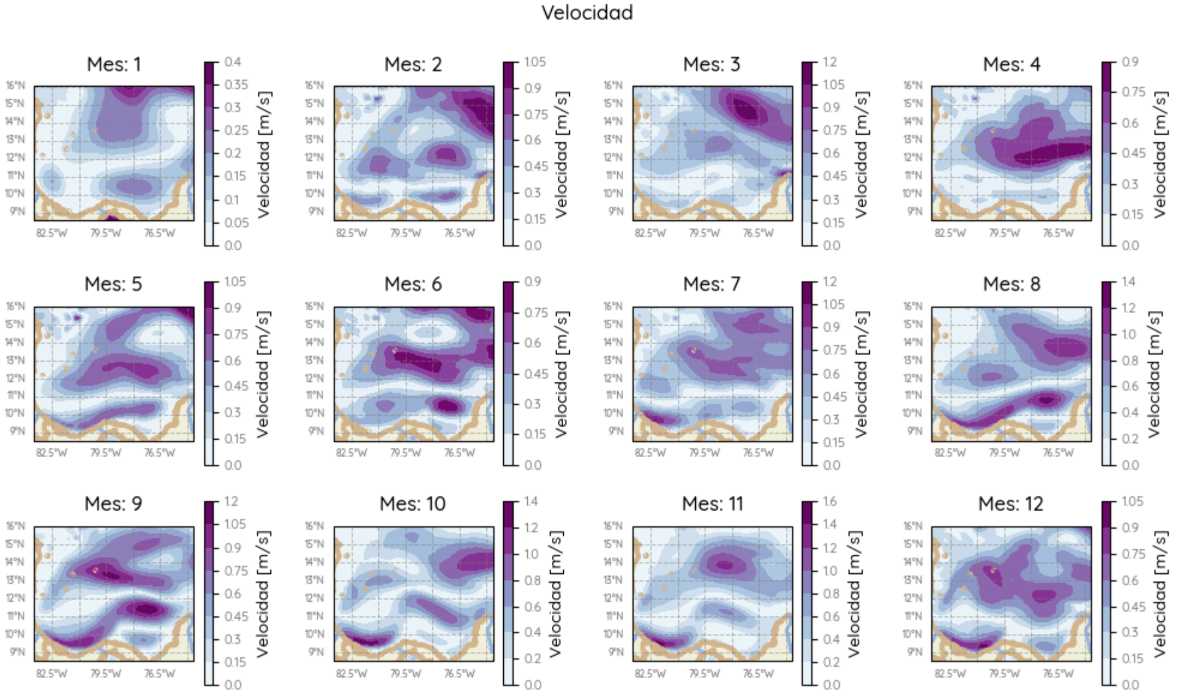
R2 0.92



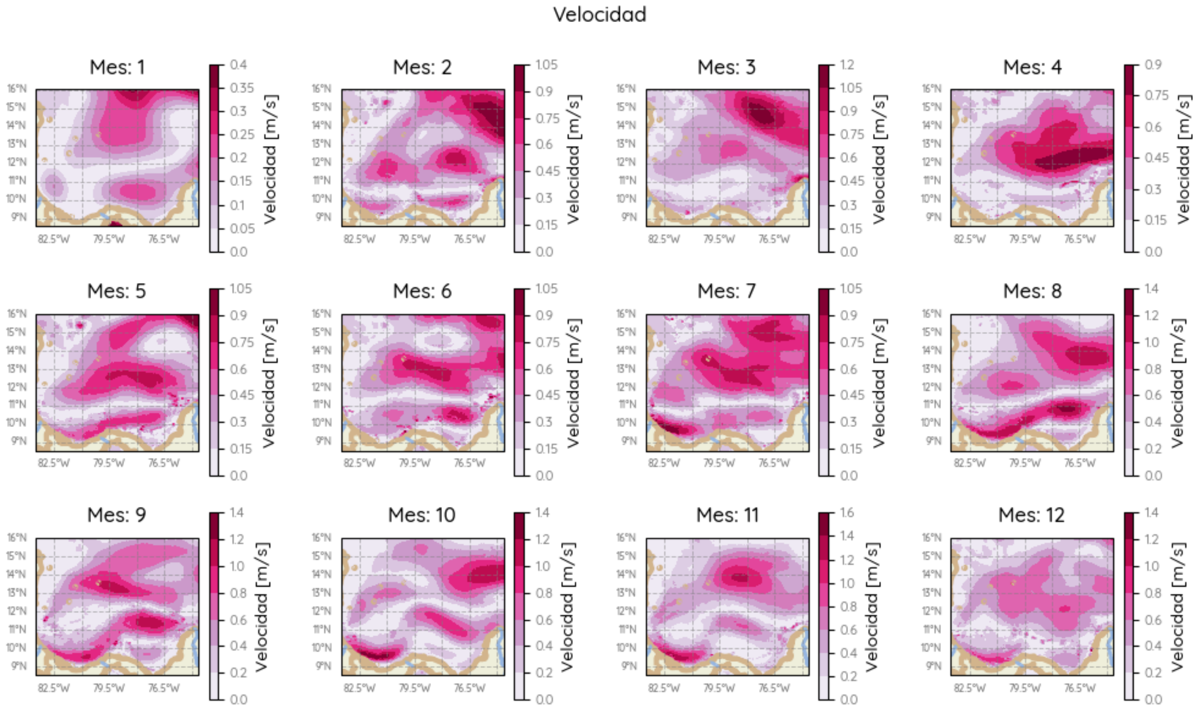


Patrones de Velocidad con KNN

Y : variable independiente



Predicción con KNN



Aplicación de Métodos- SVM

Evaluación del desempeño del modelo

```
from sklearn.svm import LinearSVR
kfold = ShuffleSplit(n_splits=5)
model = LinearSVR()
results = cross_val_score(model, X_train_sd.iloc[:,1:], y_train1, cv=kfold)
print(results.mean())
print(results.std())
```

✓ 36.7s

```
0.13139623741992826
0.003890282454871649
```

2

K-FOLD

Evaluar el rendimiento general del modelo sin entrar en detalle.

Selección de hiperparámetro C

Estandarizando los Features:

↑ C ↑ Alta varianza (overffiting) ↓ Bajo Bias

↓ C ↓ Baja varianza ↑ Alto Bias

```
from sklearn.svm import LinearSVR
c_params = np.arange(1,100,10)
param_grid = {'C': c_params}
rsearch = RandomizedSearchCV(estimator=LinearSVR(), param_distributions=param_grid, n_iter=100, random_state=1)
rsearch.fit(X_train_sd.iloc[:,1:], y_train1)
print('Mejor R2',rsearch.best_score_)
print('Mejor estimador', rsearch.best_estimator_.C)
```

✓ 9m 45.2s Python

```
Mejor R2 0.12871066863776942
Mejor estimador 1
```


UNIVERSIDAD NACIONAL DE COLOMBIA

Aplicación de Métodos- SVM

Proceso

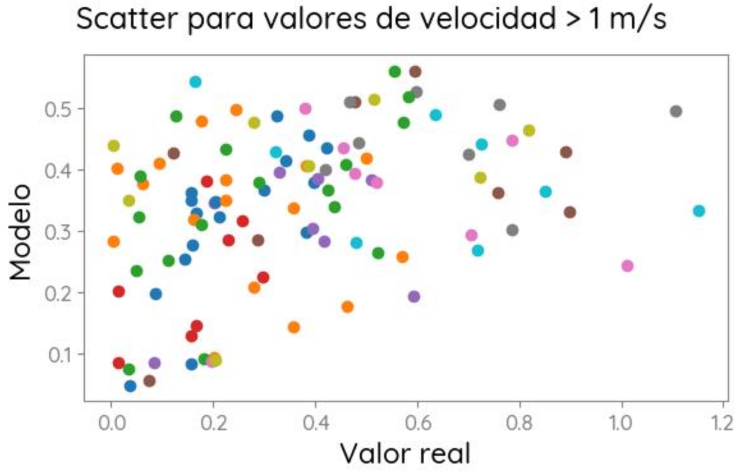
```
from sklearn.svm import LinearSVR

svr_lin = LinearSVR(C=1)
# svr_lin = SVR(kernel="linear", C=1, gamma="auto")
y_lin = svr_lin.fit(X_train_sd.iloc[:,1:],y_train1)

# Score del entrenamiento
score=svr_lin.score(X_train_sd.iloc[:,1:],y_train1)
print('R2 del entrenamiento:',score)


# Score del test
y_pred_test_svr = svr_lin.predict(X_test_sd.iloc[:,1:])
r2_test = r2_score(y_test1,y_pred_test_svr)
print('El R2 de la validación: ',r2_test)
```

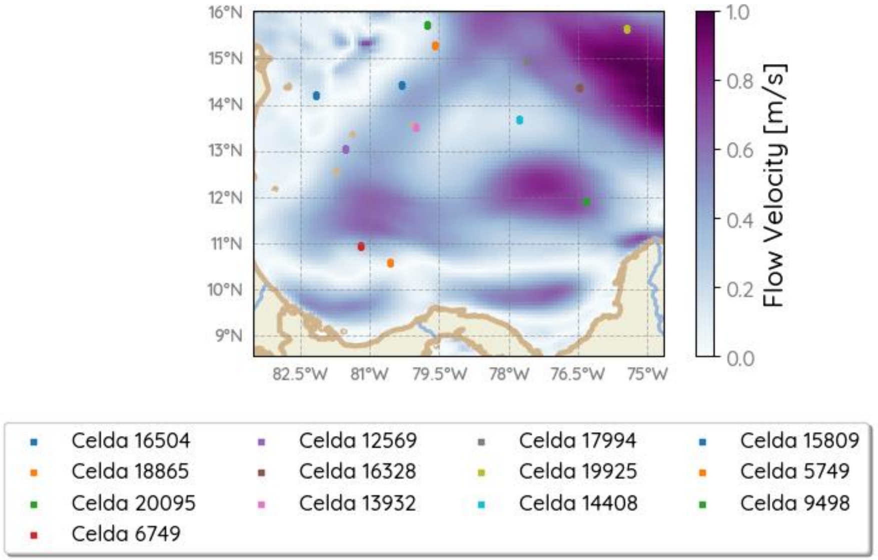
✓ 6.6s

```
R2 del entrenamiento: 0.13048822546044847
El R2 de la validación:  0.13061645633537422
```

Validación

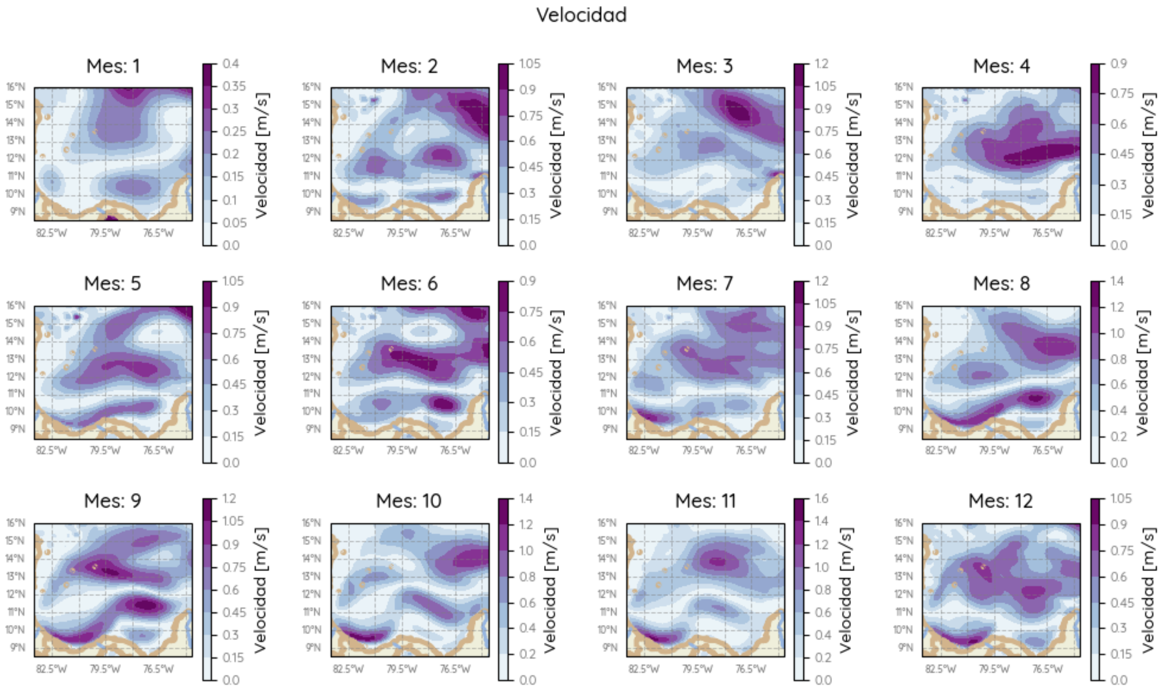
R2 0.13



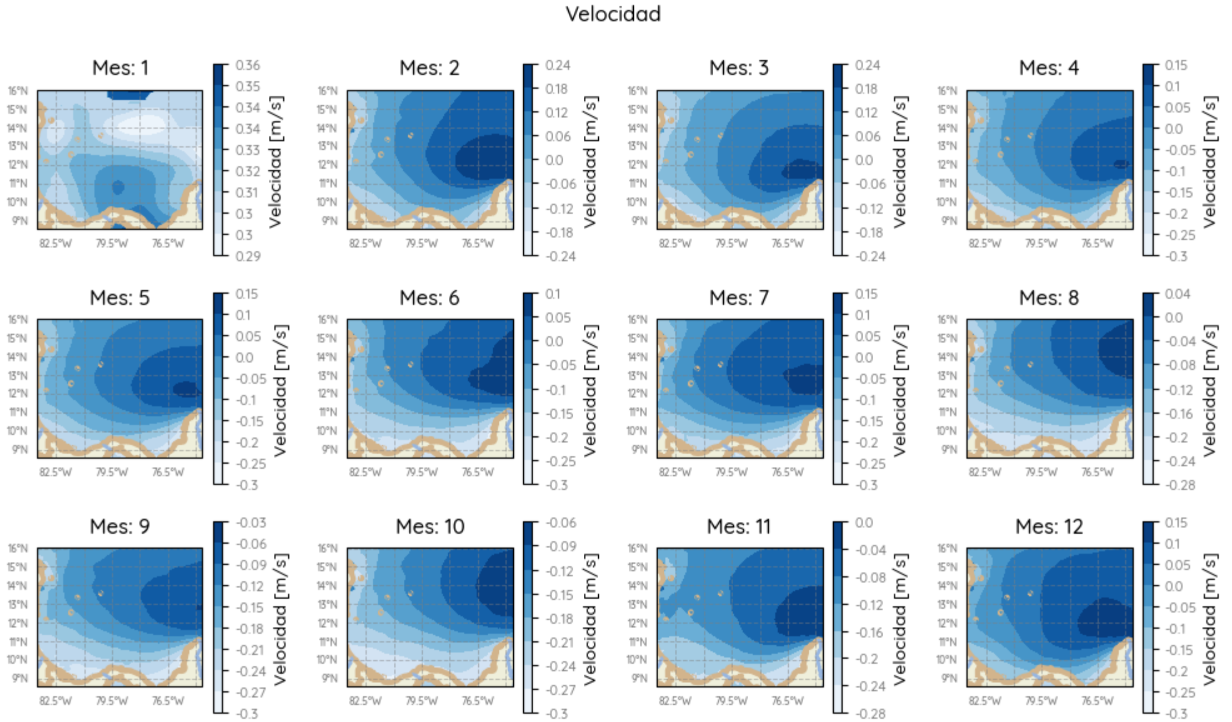


Aplicación de Métodos- SVM

Y : variable independiente



Predicción con SVR



UNIVERSIDAD NACIONAL DE COLOMBIA

Aplicación de Métodos- RAMDON FOREST

Evaluación del desempeño del modelo

```python
# Evaluación general del desempeño del modelo  con KFold
kfold = KFold(n_splits=5, shuffle= True,random_state=1)
model = RandomForestRegressor()
results1 = cross_val_score(model, X_train.iloc[:,1:], y_train1, cv=kfold, scoring='r2') # Obtengo la metrica R2
print('Resultados para la variable Y1')
print(results1, '\n' )
print(results1.mean())
print(results1.std())
```

Python

```
Resultados para la variable Y1
[0.95436978 0.95271959 0.95417483 0.95745627 0.95643969]

0.9550320304126885
0.0016958293958169108
```

```python
from sklearn.ensemble import RandomForestRegressor
# Evaluación general del desempeño del modelo
# Validación cruzada
results=cross_validate(RandomForestRegressor(), X.iloc[:,1:], Y1, return_train_score=True, cv=5)
results
```

✓ 7m 10.8s

```
{'fit_time': array([92.02941251, 86.07804728, 76.96924305, 77.05657125, 76.81329918]),
 'score_time': array([0.42453766, 0.35500216, 0.42325997, 0.45566773, 0.46896243]),
 'test_score': array([ 0.16784284,  0.04835069, -0.19511127, -0.68669327, -0.15487245]),
 'train_score': array([0.99458028, 0.99517407, 0.99618863, 0.99557515, 0.99593637])}
```

2

K-FOLD

Evaluar el rendimiento general del modelo sin entrar en detalle.

Selección de hiperparámetros

Se observó un gran gasto computacional:
Tiempo de ejecución > 1h

```python
# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf}

# search across 100 different combinations, and use all available cores
rf_random = RandomizedSearchCV(estimator=RandomForestRegressor(), param_distributions=random_grid,
                               n_iter = 10, scoring='neg_mean_absolute_error',
                               cv = 3, verbose=2, random_state=42, n_jobs=-1)


rf_random.fit(X_train.iloc[:,1:], y_train1)
print('Mejor R2',rf_random.best_score_)
print('Mejores hiperparametros',rf_random.best_params_)
```

✓ 58m 2.5s

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
Mejor R2 -0.028374386503040458
Mejores hiperparametros {'n_estimators': 385, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_features': 'auto', 'max_depth': 90}
```

Aplicación de Métodos- REDES NEURONALES

Evaluación del desempeño del modelo

```
from sklearn.neural_network import MLPRegressor
# Evaluación general del desempeño del modelo
# Validación cruzada
results=cross_validate(MLPRegressor(), X.iloc[:,1:], Y1, return_train_score=True, cv=5)
results
```

✓ 8m 9.9s

```
{'fit_time': array([115.87230015, 122.59377503, 121.87337351,  45.11284471,
         82.86115909]),
 'score_time': array([0.08034158, 0.05770755, 0.05479503, 0.06411147, 0.056216  ]),
 'test_score': array([-2.49121806e+01,  1.24692780e-01,  1.37279332e-02,  1.37102734e-02,
        -1.25599761e-01]),
 'train_score': array([0.28912138, 0.35194627, 0.43386075, 0.39022356, 0.30812313])}
```

2

K-FOLD

Evaluar el rendimiento general del modelo sin entrar en detalle.

Selección de hiperparámetros

Se observó un gran gasto computacional: Tiempo de ejecución > 1h

```
from sklearn.neural_network import MLPRegressor
# Capas ocultas
hidden_layer_sizes= np.arange(10, 100, 10)
# Función de activación
activation=['logistic', 'tanh' , 'relu']
# Solver
solver =['lbfgs', 'sgd', 'adam']

# Create the random grid
random_grid = {'hidden_layer_sizes': hidden_layer_sizes,
               'activation': activation,
               'solver': solver}
# search across 100 different combinations, and use all available cores
rf_random = RandomizedSearchCV(estimator=MLPRegressor(), param_distributions=random_grid,
                               n_iter = 10, scoring='neg_mean_absolute_error',
                               cv = 3, verbose=2, random_state=42, n_jobs=-1)
rf_random.fit(X_train.iloc[:,1:], y_train1)
print('Mejor R2',rf_random.best_score_)
print('Mejores hiperparametros',rf_random.best_params_)
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
Mejor R2 -0.18339299571501808
Mejores hiperparametros {'solver': 'lbfgs', 'hidden_layer_sizes': 70, 'activation': 'logistic'}
```


UNIVERSIDAD NACIONAL DE COLOMBIA

Modelo Seleccionado

Modelo	R2 Testo
Regresión Lineal	**0.151**
Regresión Lineal (Variables Categóricas)	**0.31**
KNN	**0.92**
Suport Vector Regressor	**0.13**
Modelos Ensamblados	**Computacionalmente altos**

Conclusiones

- El conocimiento profundo de los **features y la variable a predicir** es importante a la hora de elegir cual modelo se ajusta con la métrica deseada.
- SVM, en sus modelos para regresión está limitado a la cantidad de muestras.
- En general el desempeño de los modelos **no paramétricos** mostraron **métricas más altas**.
- Se debe probar el modelo de ML, en una región diferente.
- A grandes rasgos los modelos ensamblados son computacionalmente altos.

GRACIAS POR SU ATENCIÓN

Referencias

- Escobar, Carlos A., Liliana Velásquez, and Federico Posada. 2015. "Marine Currents in the Gulf of Urabá, Colombian Caribbean Sea." Journal of Coastal Research 31(6):1363–74.
- Liu, Y., & Weisberg, R. H. (2005). Patterns of ocean current variability on the West Florida Shelf using the self-organizing map. Journal of Geophysical Research: Oceans, 110(6), 1–12. https://doi.org/10.1029/2004JC002786
- Ambroise, C., G. Seze, F. Badran, and S. Thiria (2000), Hierarchical clustering of self-organizing maps for cloud classification, Neurocomputing, 30, 47 – 52
- Jirakittayakorn, A., Kormongkolkul, T., Vateekul, P., Jitkajornwanich, K., & Lawawirojwong, S. (2017). Temporal kNN for short-Term ocean current prediction based on HF radar observations. Proceedings of the 2017 14th International Joint Conference on Computer Science and Software Engineering, JCSSE 2017. https://doi.org/10.1109/JCSSE.2017.8025921