INTRODUCCIÓN AL MACHINE LEARNING

PRECIOS DE LA VIVIENDA - TÉCNICAS AVANZADAS DE REGRESIÓN

1° CHALLENGE

EQUIPO

DIANA LLAMOCA

BRIGITTE BERNAL

MASSIEL COLLA

STEPHANY TORIBIO

RESUMEN DE CONTENIDOS

I. MÉTOD0

II. FRAMEWORKS Y MECANISMOS DE ENTRENAMIENTO

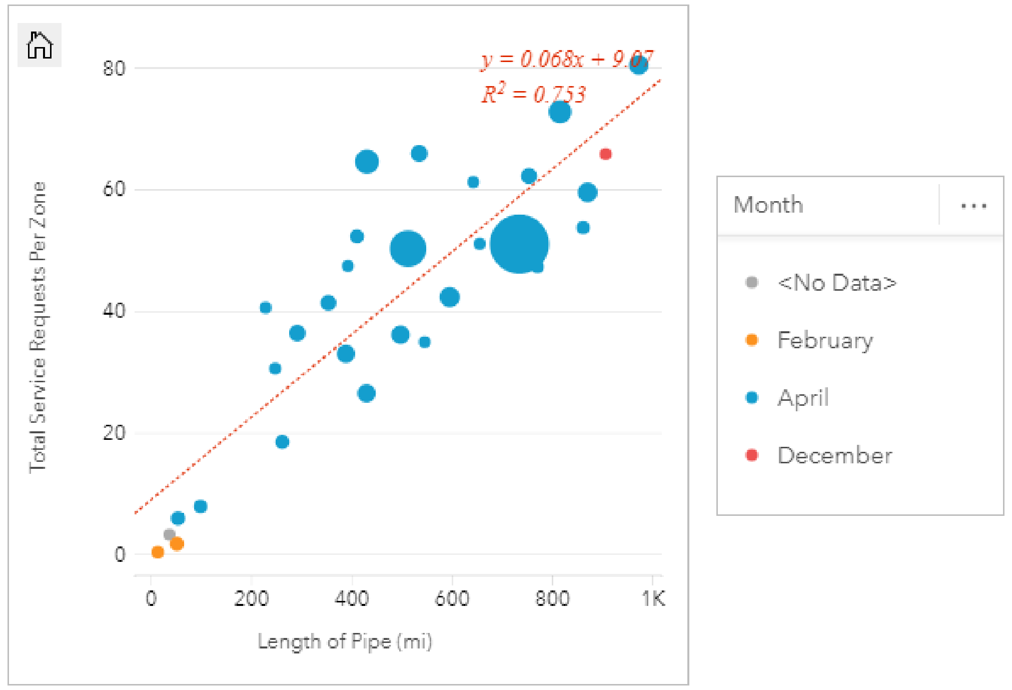
III. RESULTADOS

IV. MEJORAS FUTURAS

DESAFÍO 1

INTRODUCCIÓN

INTRODUCCIÓN



MÉTODOS

MANEJO DE ARCHIVOS

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O
(e.g. pd.read_csv)
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
   for filename in filenames:
      print(os.path.join(dirname, filename))
train=pd.read_csv("/kaggle/input/house-prices-
advanced-regression-techniques/train.csv")

train.index=train["Id"]
train.drop("Id",axis=1,inplace=True)
train.info()
#Vamos a eliminar las columnas con demasiados
valores faltantes
train[["Alley","PoolQC","Fence","MiscFeature"]].isnull().s
um()
train.drop(["Alley","PoolQC","Fence","MiscFeature"],
axis=1, inplace=True)
train.shape
Num=[col for col in train.columns if
train[col].dtype!="object" and col!="SalePrice"]
Cat=[col for col in train.columns if
train[col].dtype=="object"]
```

OS module

walk()

join()

PREPROCESAMIENTO

scikit learn machine learning in Python

preprocessing.impute.SimpleImputer()

preprocessing.StandardScaler()

preprocessing.OrdinalEncoder()

linear_model.ElasticNet()

```
train_feat=train[Num]
train_target=train["SalePrice"]
from sklearn import impute
SI=impute.SimpleImputer(strategy="mean")
train_feat_si=pd.DataFrame(SI.fit_transform(tr
ain_feat))
train_feat_si.index=train_feat.index
train_feat_si.columns=train_feat.columns
#Colocando a la misma escala los datos
numéricos, puesto que los valores de los features
están en escalas distintas
from sklearn import preprocessing
SE=preprocessing.StandardScaler()
train_feat_t=pd.DataFrame(SE.fit_transform(tra
in_feat_si))
train_feat_t.index=train_feat_si.index
train_feat_t.columns=train_feat_si.columns
```

MÉTODOS

MANEJO DE DATOS

MODELO DE REGRESIÓN

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O
(e.g. pd.read_csv)
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
train=pd.read_csv("/kaggle/input/house-prices-
advanced-regression-techniques/train.csv")

train.index=train["Id"]
train.drop("Id",axis=1,inplace=True)
train.info()
#Vamos a eliminar las columnas con demasiados
valores faltantes
train[["Alley","PoolQC","Fence","MiscFeature"]].isnull().s
um()
train.drop(["Alley","PoolQC","Fence","MiscFeature"],
axis=1, inplace=True)
train.shape
Num=[col for col in train.columns if
train[col].dtype!="object" and col!="SalePrice"]
Cat=[col for col in train.columns if
train[col].dtype=="object"]
```

Pandas

read_csv()

drop()

isnull().sum()

DataFrame()

scikits learn machine learning in Python

ElasticNet()

fit()

predict()

```
test_features_final=pd.concat([test_
features_esc,test_feat_cat_f],axis=1)
test_target_final=test_target["SalePri
ce"]
predicciones12=RLR.predict(test_fea
tures_final)
output12=pd.DataFrame({'Id':output_
submission_12.Id':output_submission
_12.Id, 'SalePrice':predicciones12})
output12.to_csv('submission_new_12.c
sv',index=False)
output12=pd.DataFrame({'Id':output_
submission_12.Id,
'SalePrice':predicciones12})
output12.to_csv('submission_new_12.c
sv',index=False)
```

FRAMEWORKS

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O
(e.g. pd.read_csv)
import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
train=pd.read_csv("/kaggle/input/house-prices-
advanced-regression-techniques/train.csv")

train.index=train["Id"]
train.drop("Id",axis=1,inplace=True)
train.info()
#Vamos a eliminar las columnas con demasiados
valores faltantes
train[["Alley","PoolQC","Fence","MiscFeature"]].isnull().s
um()
train.drop(["Alley","PoolQC","Fence","MiscFeature"],
axis=1, inplace=True)
train.shape
Num=[col for col in train.columns if
train[col].dtype!="object" and col!="SalePrice"]
Cat=[col for col in train.columns if
train[col].dtype=="object"]
```

Pandas

OS module

scikits learn

machine learning in Python

```
train_feat=train[Num]
train_target=train["SalePrice"]
from sklearn import impute
SI=impute.SimpleImputer(strategy="mean")
train_feat_si=pd.DataFrame(SI.fit_transform(tr
ain_feat))
train_feat_si.index=train_feat.index
train_feat_si.columns=train_feat.columns
#Colocando a la misma escala los datos
numéricos, puesto que los valores de los features
están en escalas distintas
from sklearn import preprocessing
SE=preprocessing.StandardScaler()
train_feat_t=pd.DataFrame(SE.fit_transform(tra
in_feat_si))
train_feat_t.index=train_feat_si.index
train_feat_t.columns=train_feat_si.columns
```

FRAMEWORKS

```
#Features categóricas:
train_feat_cat=train[Cat]
SI_Cat=impute.SimpleImputer(strategy="most_frequent")
train_feat_c_n=pd.DataFrame(SI_Cat.fit_transform(train_feat_cat))
train_feat_c_n.index=train_feat_cat.index
train_feat_c_n.columns=train_feat_cat.columns
from sklearn import preprocessing
OE=preprocessing.OrdinalEncoder()
train_feat_c_f=pd.DataFrame(OE.fit_transform(train_feat_c_n))
train_feat_c_f.index=train_feat_c_n.index
train_feat_c_f.columns=train_feat_c_n.columns
train_features_final=pd.concat([train_feat_t,train_feat_c_f],axis=1)
train_target_final=train["SalePrice"]
```

Pandas

scikits learn
machine learning in Python

```
#Entrenamos al modelo
from sklearn import linear_model
#Regresión lineal ElasticNet
RLR=linear_model.ElasticNet() #Dejamos el valor de alpha por defecto
#Entrenando al modelo
RLR.fit(train_features_final,train_target_final)
test_features=pd.read_csv("/kaggle/input/house-prices-advanced-regression-techniques/test.csv")
test_target=pd.read_csv("/kaggle/input/house-prices-advanced-regression-techniques/sample_submission.csv")

test_features.index=test_features["Id"]
test_features.drop("Id",axis=1,inplace=True)

#Elegimos la data numérica del test
test_feat_num=test_features[Num]
output_submission_12=test_target.copy()
######

test_target.index=test_target["Id"]
test_target.drop("Id",axis=1,inplace=True)
test_features_f=pd.DataFrame(SI.transform(test_feat_num))
test_features_f.index=test_feat_num.index
test_features_f.columns=test_feat_num.columns
```

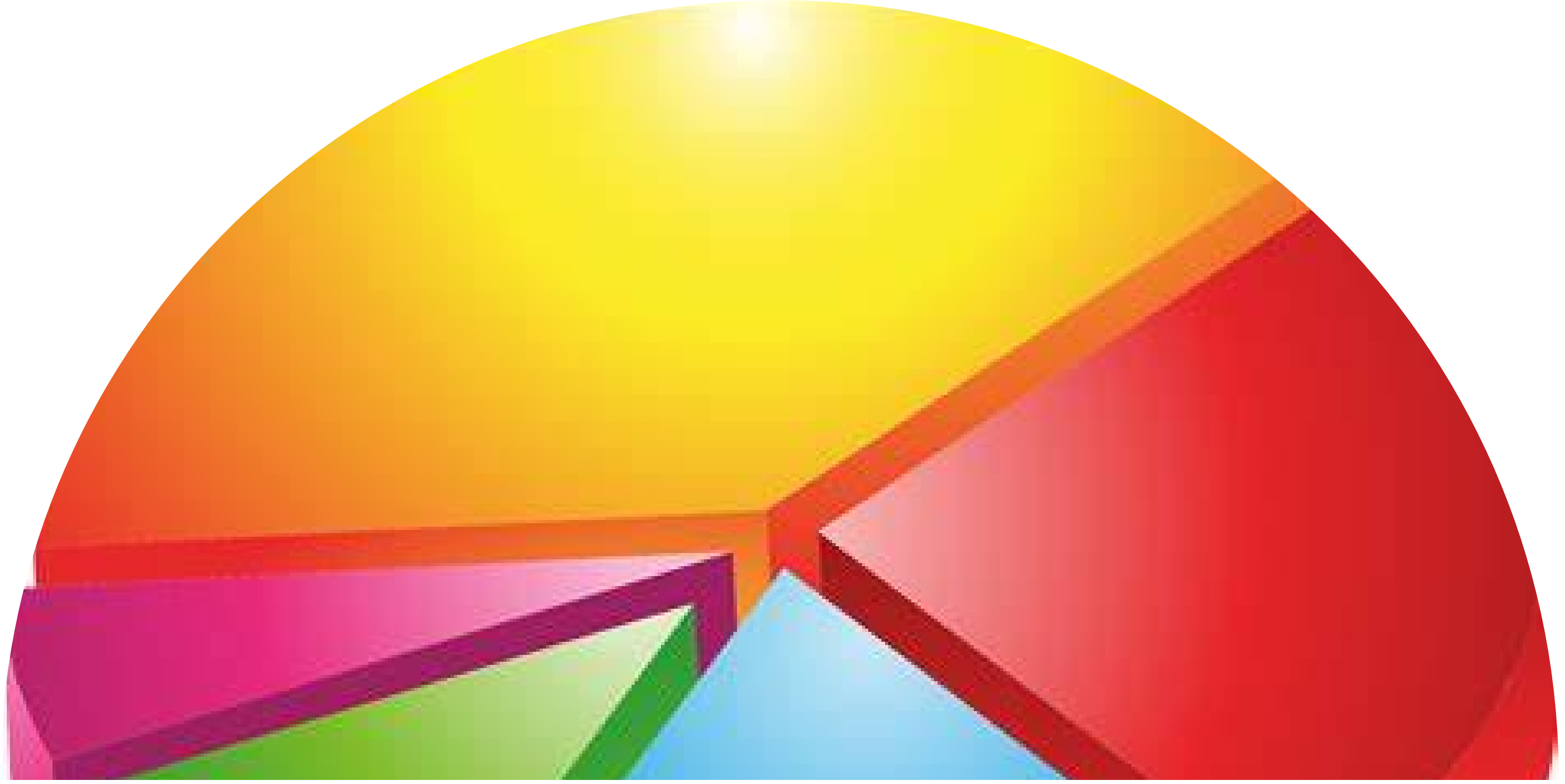
FRAMEWORKS

```
#Al ya no haber valores nulos, se igualará la escala de los valores numéricos
test_features_esc=pd.DataFrame(SE.transform(test_features_f))
test_features_esc.index=test_features_f.index
test_features_esc.columns=test_features_f.columns

#Ahora, la data categórica
test_features_cat=test_features[Cat]

test_feat_cat_n=pd.DataFrame(SI_Cat.transform(test_features_cat))
test_feat_cat_n.index=test_features_cat.index
test_feat_cat_n.columns=test_features_cat.columns
test_feat_cat_f=pd.DataFrame(OE.transform(test_feat_cat_n))
test_feat_cat_f.index=test_feat_cat_n.index
test_feat_cat_f.columns=test_feat_cat_n.columns
```

Pandas

scikits learn machine learning in Python

```
test_features_final=pd.concat([test_features_esc,test_feat_cat_f],axis=1)
test_target_final=test_target["SalePrice"]
predicciones12=RLR.predict(test_features_final)
output12=pd.DataFrame({'Id':output_submission_12.Id':output_submission_12.Id, 'SalePrice':predicciones12})
output12.to_csv('submission_new_12.csv',index=False)
output12=pd.DataFrame({'Id':output_submission_12.Id,
'SalePrice':predicciones12})
output12.to_csv('submission_new_12.csv',index=False)
```



RESULTADOS

RESULTADOS DEL MODELO DE REGRESIÓN LINEAL

RESULTADOS

RMSE (EN LA DATA DE TEST)

```
RMSE_test: 65054.557662585736
```

KAGGLE

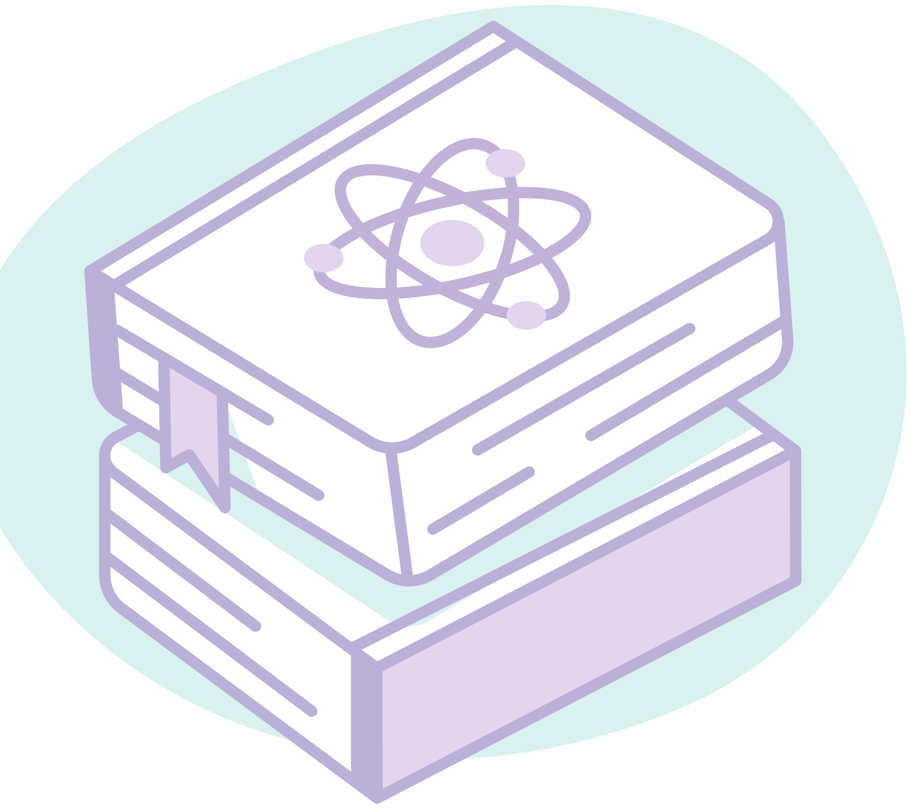
Public Score	Best Score
0.15646	0.15646 V1

RMSE EN LA DATA DE TRAIN Y TEST

```
#Métrica RMSE
import math
from sklearn import metrics

print("RMSE_train:",math.sqrt(metrics.mean_squared_error(EN.predict(train_features_final),train_target_final)))

print("RMSE_test:",math.sqrt(metrics.mean_squared_error(EN.predict(test_features_final),test_target)))


#Sin embargo, el puntaje final de Kaggle (el Public Score) es 0.15646
```

```
RMSE_train: 32350.405832316555
RMSE_test: 65054.557662585736
```

Por ello, debemos saber que sí se pueden hacer posibles mejoras futuras con el objetivo de mejorar el ‘performance’ del modelo de ML.



POSIBLES MEJORAS FUTURAS

- Exploración de Datos:
- Operaciones destructivas
- Optimización de Hiperparámetros
- Validación Cruzada y Evaluación de Modelos
- Considerar otras bibliotecas
- Selección de Características

¡MUCHAS GRACIAS!

CONÓCENOS

@SOFTWARE_GIRLSTEAM