



Ingeniería en Informática
INVESTIGACIÓN DE OPERACIONES

Profesora Karla López



Gipsy Anaya
gipsyanaya@gmail.com
C.i: 27.966.466

Ciencias de datos con Python

Proyecto

Enunciado del ejercicio

El hundimiento del Titanic es uno de los naufragios más infames de la historia.

El 15 de abril de 1912, durante su viaje inaugural, el RMS Titanic ampliamente considerado "insumergible" se hundió después de chocar con un iceberg. Desafortunadamente, no había suficientes botes salvavidas para todos a bordo, lo que resultó en la muerte de 1502 de los 2224 pasajeros y tripulantes. Si bien hubo algún elemento de suerte involucrado en la supervivencia, parece que algunos grupos de personas tenían más probabilidades de sobrevivir que otros.

En este desafío, le pedimos que construya un modelo predictivo que responda a la pregunta: "¿qué tipo de personas tenían más probabilidades de sobrevivir?" utilizando datos de pasajeros (es decir, nombre, edad, sexo, clase socioeconómica, etc.)

Objetivo Principal

Se nos presenta un texto dando una descripción clara y concisa del ejercicio del Titanic en Kaggle, el cual nos indica que debemos desarrollar un modelo de aprendizaje automático utilizando las técnicas de ciencias de datos, en lenguaje Python, que pueda predecir si un pasajero sobrevivió o no en función de características de los pasajeros.

Programas utilizados: Visual Studio Code
a través de Anaconda



Metodología : Paso a Paso

1) Importar los paquetes o librerías necesarias:

```
1  # Importar librerías
2  import pandas as pd
3  import numpy as np
4
5  from sklearn.model_selection import train_test_split
6  from sklearn.linear_model import LogisticRegression
7  from sklearn.svm import SVC
8  from sklearn.neighbors import KNeighborsClassifier
```

En primer lugar, se importan las librerías necesarias para manejar y manipular los datos, y aplicar técnicas de aprendizaje automático. En este caso, se importan Pandas para manejar los datos, NumPy para realizar cálculos matemáticos, y scikit-learn para aplicar técnicas de aprendizaje automático.

2) Cargar los datos de los archivos proporcionados por kaggle

```
10 # Cargar los datos de los archivos proporcionados por Kaggle
11 train = r'c : \Users\elton\Downloads\Ale\6to semestre\investigacion de operaciones\Ciencias de datos en Python\train.csv'
12 test = r'c : \Users\elton\Downloads\Ale\6to semestre\investigacion de operaciones\Ciencias de datos en Python\test.csv'
13
14 train_file = 'train.csv'
15 test_file = 'test.csv'
16 df_train = pd.read_csv(train_file)
17 df_test = pd.read_csv(test_file)
```

Se cargan los datos de los archivos proporcionados por Kaggle utilizando la función `read_csv` de Pandas y se crean dos dataframes: `df_train` y `df_test`.

3) Verificar los datos:

```
19 # Verificar los datos
20 print("\nDatos de entrenamiento:\n ")
21 print(df_train.head())
22 print(df_train.info())
23 print(df_train.isnull().sum())
24
25 print("\nDatos de prueba:\n")
26 print(df_test.head())
27 print(df_test.info())
28 print(df_test.isnull().sum())
```

Se verifica la calidad de los datos utilizando las funciones head, info y isnull. Se observa que hay valores faltantes en las columnas 'Age', 'Embarked' y 'Fare' en el conjunto de pruebas.

4)Preprocesamiento de datos

```
30 # Preprocesamiento de datos
31 df_train.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)
32 df_test.drop(['PassengerId', 'Name', 'Ticket', 'Cabin'], axis=1, inplace=True)
33 df_train['Sex'] = df_train['Sex'].map({'male': 1, 'female': 0})
34 df_test['Sex'] = df_test['Sex'].map({'male': 1, 'female': 0})
35 df_train['Embarked'] = df_train['Embarked'].map({'S': 0, 'C': 1, 'Q': 2})
36 df_test['Embarked'] = df_test['Embarked'].map({'S': 0, 'C': 1, 'Q': 2})
37 df_train['Age'].fillna(df_train['Age'].mean(), inplace=True)
38 df_test['Age'].fillna(df_test['Age'].mean(), inplace=True)
39 df_train['Age'] = pd.cut(df_train['Age'], bins=[0, 8, 15, 18, 25, 40, 60, 100], labels=['1', '2', '3', '4', '5', '6', '7'])
40 df_test['Age'] = pd.cut(df_test['Age'], bins=[0, 8, 15, 18, 25, 40, 60, 100], labels=['1', '2', '3', '4', '5', '6', '7'])
41 df_train.dropna(inplace=True)
```

Recordando que estamos utilizando Pandas, se realiza el preprocesamiento de los datos, eliminando las columnas 'PassengerId', 'Name', 'Ticket' y 'Cabin' utilizando la función drop. Se convierten los valores categóricos 'male' y 'female' de la columna 'Sex' en valores numéricos utilizando la función map. Se convierten los valores categóricos 'S', 'C' y 'Q' de la columna 'Embarked' en valores numéricos utilizando la función map. Se reemplazan los valores faltantes en la columna 'Age' con la media de la columna utilizando la función fillna. Se agrupan las edades en siete grupos utilizando la función cut, y se eliminan las filas con valores faltantes utilizando la función dropna.

5)Entrenamiento y Prueba de datos

```
43 # Entrenamiento y Prueba de datos
44 X = df_train.drop('Survived', axis=1)
45 y = df_train['Survived']
46 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Se separan los datos en conjuntos de entrenamiento (80%) y prueba (20%) utilizando la función `train_test_split` de `scikit-learn`. Se definen las variables `X` y `y` para el conjunto de entrenamiento, donde `X` contiene todas las columnas excepto la columna 'Survived', y `y` contiene la columna 'Survived'.

6)Crear Modelos de aprendizaje automático

```
49 #Regresión Logística
50 log_reg = LogisticRegression()
51 log_reg.fit(X_train, y_train)
52 y_pred = log_reg.predict(X_test)
53 print("\nPrecisión Regresión Logística:\n"),
54 print (log_reg.score(X_test, y_test))
55
56 #SVM
57 svm = SVC()
58 svm.fit(X_train, y_train)
59 y_pred = svm.predict(X_test)
60 print("\nPrecisión SVM:\n")
61 print (svm.score(X_test, y_test))
62
63 #K-Nearest Neighbors
64 knn = KNeighborsClassifier()
65 knn.fit(X_train, y_train)
66 y_pred = knn.predict(X_test)
67 print("\nPrecisión KNN:\n")
68 print(knn.score(X_test, y_test))
```

Se crean tres modelos de aprendizaje automático: Regresión Logística, SVM y K-Nearest Neighbors. Se ajustan los modelos utilizando la función `fit` y se realizan las predicciones utilizando la función `predict`. Se evalúa la precisión de cada modelo utilizando la función `score` en los datos de prueba. Se imprime la precisión de cada modelo.

7) Predicción con modelos

```
70 #Predicción con modelos
71 test_ids = df_test['PassengerId']
72 test_data = df_test.drop('PassengerId', axis=1)
73 test_data.fillna(test_data.mean(), inplace=True)
74
75 log_reg_pred = log_reg.predict(test_data)
76 svm_pred = svm.predict(test_data)
77 knn_pred = knn.predict(test_data)
78
79 log_reg_output = pd.DataFrame({'PassengerId': test_ids, 'Survived': log_reg_pred})
80 svm_output = pd.DataFrame({'PassengerId': test_ids, 'Survived': svm_pred})
81 knn_output = pd.DataFrame({'PassengerId': test_ids, 'Survived': knn_pred})
82
```

Se realiza la predicción de la supervivencia de los pasajeros en el conjunto de pruebas utilizando los modelos creados anteriormente. Se crea un dataframe para cada modelo con las columnas 'PassengerId' y 'Survived'.

8) Se imprimen los datos

```
83 print("\n\nPredicción Regresión Logística:")
84 print(log_reg_output.head())
85 print("\n\nPredicción SVM:")
86 print(svm_output.head())
87 print("\n\nPredicción KNN:")
88 print(knn_output.head())
```

Se imprimen las predicciones de cada modelo utilizando la función head.

Resultados obtenidos
en la terminal de Visual
Code Studio

Datos de entrenamiento

```
PS C:\Users\elton\Downloads\Ale\6to semestre\investigacion de operaciones\Ciencias de datos en Python> & C:/Users/elton/anaconda3/python.exe "c:/Users/elton/Downloads/Ale/6to semestre/investigacion de operaciones/Ciencias de datos en Python/Titanic.py"
```

Datos de entrenamiento:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890

Tipos de datos

```
Data columns (total 12 columns):
#   Column          Non-Null Count  Dtype
---  -
0   PassengerId      891 non-null    int64
1   Survived          891 non-null    int64
2   Pclass            891 non-null    int64
3   Name              891 non-null    object
4   Sex               891 non-null    object
5   Age               714 non-null    float64
6   SibSp             891 non-null    int64
7   Parch            891 non-null    int64
8   Ticket            891 non-null    object
9   Fare              891 non-null    float64
10  Cabin             204 non-null    object
11  Embarked          889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Datos faltantes

```
None
PassengerId      0
Survived          0
Pclass           0
Name             0
Sex              0
Age             177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin           687
Embarked         2
dtype: int64
```

Datos de prueba

```
   PassengerId  Pclass                               Name  Sex  Age  SibSp  Parch  Ticket   Fare  Cabin  Embarked
0            892      3                    Kelly, Mr. James  male  34.5     0     0   330911   7.8292   NaN      Q
1            893      3      Wilkes, Mrs. James (Ellen Needs) female  47.0     1     0   363272   7.0000   NaN      S
2            894      2      Myles, Mr. Thomas Francis   male  62.0     0     0   240276   9.6875   NaN      Q
3            895      3                    Winz, Mr. Albert  male  27.0     0     0   315154   8.6625   NaN      S
4            896      3  Hirvonen, Mrs. Alexander (Helga E Lindqvist) female  22.0     1     1  3101298  12.2875   NaN      S
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
```

Tipos de datos

```
Data columns (total 11 columns):
#   Column      Non-Null Count  Dtype
---  -
0   PassengerId  418 non-null    int64
1   Pclass       418 non-null    int64
2   Name         418 non-null    object
3   Sex          418 non-null    object
4   Age          332 non-null    float64
5   SibSp        418 non-null    int64
6   Parch        418 non-null    int64
7   Ticket       418 non-null    object
8   Fare         417 non-null    float64
9   Cabin        91 non-null     object
10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
```

Datos faltantes

```
None
PassengerId      0
Pclass           0
Name             0
Sex              0
Age             86
SibSp            0
Parch            0
Ticket           0
Fare             1
Cabin           327
Embarked         0
dtype: int64
```


Conclusión

En el ejercicio, se utilizan tres modelos de aprendizaje automático para predecir la supervivencia de los pasajeros del Titanic: Regresión Logística, SVM y K-Nearest Neighbors. Después de ajustar cada modelo y realizar las predicciones en el conjunto de prueba, se evalúa la precisión de cada modelo utilizando la función score en los datos de prueba.

En general, el ejercicio proporciona un buen ejemplo de cómo utilizar diferentes modelos de aprendizaje automático para resolver un problema de clasificación y cómo evaluar la precisión de cada modelo

Gracias por su tiempo



UNEG

> Ingeniería en Informática

>> Investigación de Operaciones

>>> Tema

>>>> Gipsy Anaya