Bochra CHEMAM

Data science

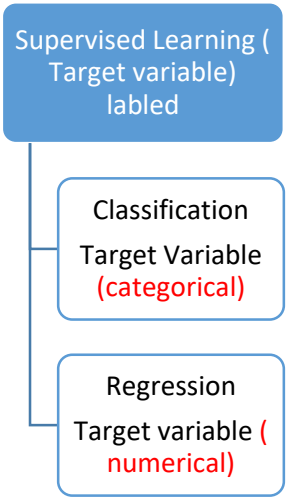
"The best way to learn data science is to apply data science."

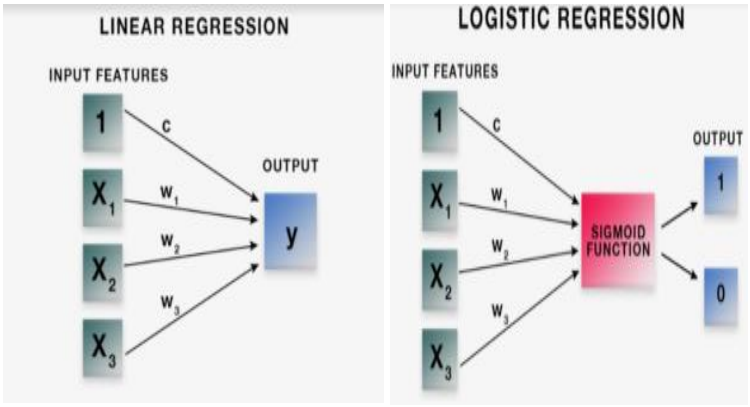
Activity 6 14/04/2021

For this Activity session we will:

- Work through a classification mini project (The hello world for machine learning).
- Visualize the data
- Evaluate the algorithms

Before we start I will give you a brief summary about classification as a type of supervised Machine learning methods :





- W**e will be able to build a logistic** regression:

a. The logistic regression algorithm helps us to find the best fit **logistic function** to describe the relationship between X and y.
b. For the classic logistic regression, y is a **binary** variable with two possible values, such as win/loss, good/bad
c. **When new observations come in, we can use its input variables and the logistic relationship to** predict the **probability** of the new case belonging to class y = 1

$$P(y = 1 \mid X) = p$$

d. Linear regression is estimated using **Ordinary Least Squares (OLS)** while logistic regression is estimated using **Maximum Likelihood Estimation (MLE)** approach.

e. Some common **applications** for logistic regression include:
   - fraud detection.
   - customer churn prediction.
   - cancer diagnosis.

f. Logistic Regression Assumptions

* Binary logistic regression requires the dependent variable to be binary.
* For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
* Only the meaningful variables should be included.
* The independent variables should be independent of each other. That is, the model should have little or no multicollinearity.
* The independent variables are linearly related to the log odds.
* Logistic regression requires quite large sample sizes.

Keeping the above assumptions in mind, let's look at our dataset.

I. Load dataset

url = ' https://raw.githubusercontent.com/BigDataGal/Python-for-Data-Science/master/titanic-train.csv '

II. VARIABLE DESCRIPTIONS

- Survived - Survival (0 = No; 1 = Yes)
- Pclass - Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
- Name - Name
- Sex - Sex
- Age - Age
- SibSp - Number of Siblings/Spouses Aboard
- Parch - Number of Parents/Children Aboard
- Ticket - Ticket Number
- Fare - Passenger Fare (British pound)
- Cabin - Cabin
- Embarked - Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

1. The first thing we are going to do is to read in the dataset using the Pandas' read_csv() function. We will put this data into a Pandas DataFrame, called "titanic", and name each of the columns.

2. titanic.info()

3. Checking that your target variable is binary

III. Data preprocessing

4. check for missing values by calling the isnull() method, and the sum() method off of that, to return a tally of all the True values that are returned by the isnull() method.

5. **Taking care of missing values**

   We will drop all the variables that aren't relevant for predicting survival. We should at least keep the following:

   a. **Survived** - This variable is relevant.
   b. **P-class** - Does a passenger's class on the boat affect their survivability?
   c. **Sex** - Could a passenger's gender impact their survival rate?
   d. **Age** - Does a person's age impact their survival rate?
   e. **SibSp** - Does the number of relatives on the boat (that are siblings or a spouse) affect a person's survivability? Probability
   f. **Parch** - Does the number of relatives on the boat (that are children or parents) affect a person's survivability? Probability
   g. **Fare** - Does the fare a person paid affect his survivability? Maybe - let's keep it.
   h. **Embarked** - Does a person's point of embarkation matter? It depends on how the boat was filled... Let's keep it.
   i. What about a person's name, ticket number, and passenger ID number? They're irrelevant for predicting survivability. And as you recall, the cabin variable is almost all missing values, so we can just drop all of these.

      titanic_data = titanic.drop(['PassengerId','Name','Ticket','Cabin'], 1)

6. Imputing missing values

   Let's look at how passenger age is related to their class as a passenger on the boat

   - we could say that the younger a passenger is, the more likely it is for them to be in 3rd class. The older a passenger is, the more likely it is for them to be in 1st class. So there is a loose relationship between these variables. So, let's write a function that approximates a passenger's age, based on their class.

   - From the box plot, it looks like the average age of 1st class passengers is about 37, 2nd class passengers is 29, and 3rd class passengers is 24.

- Create a function to fill out the missing values based on the relation extracted from the previous boxplot : **def age_approx()**

- **we see that there are no more null values in the age variable after applying age_approx() function .**

```
titanic_data['Age'] = titanic_data[['Age', 'Pclass']].apply(age_approx, axis=1)

titanic_data.isnull().sum()
```

7. Converting categorical variables to dummy variables

   The next thing we need to do is reformat our variables so that they work with the model. Specifically, we need to reformat the Sex and Embarked variables into numeric variables.

```
gender = pd.get_dummies(titanic_data['Sex'],drop_first=True)

embark_location = pd.get_dummies(titanic_data['Embarked'],drop_first=True)
```

8. Drop the sex and embarked using pd.drop and add the dummies var using pd.concat

```
titanic_data.drop(['Sex', 'Embarked'],axis=1,inplace=True)

titanic_dmy = pd.concat([titanic_data,gender,embark_location],axis=1)
```

9. **Checking for independence between features : .corr()**

10. **Use .iloc to split your data x and y**

11. **Split to train and test set (75% / 25%)**

12. drop dependent variables

IV. **Deploying and evaluating the model**

- **Deploying**
  1. LogReg = LogisticRegression()
  2. Fit x_train and y_train `LogReg.fit(X_train, y_train)`
  3. Predict y_pred using `y_pred=LogReg.predict(X_test)`
- **Evaluation**
  We will use confusion matrix :

is the basis of all performance metrics for models with a categorical response (such as a logistic regression). It contains the counts of each actual response-predicted response pair. In this case, where there are two possible responses (churn or not churn), there are four overall outcomes.

1. **True positive:** The passenger survived and the model predicted they would.
2. **False-positive:** The passenger didn't survive, but the model predicted they would.
3. **True negative:** The passenger didn't survive and the model predicted they wouldn't.
4. **False-negative:** The passenger survived, but the model predicted they wouldn't.

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

		Predicted Class	
		No	Yes
Observed Class	No	TN	FP
	Yes	FN	TP

TN	True Negative
FP	False Positive
FN	False Negative
TP	True Positive

Model Performance

Accuracy	= (TN+TP)/(TN+FP+FN+TP)
Precision	= TP/(FP+TP)
Sensitivity	= TP/(TP+FN)
Specificity	= TN/(TN+FP)