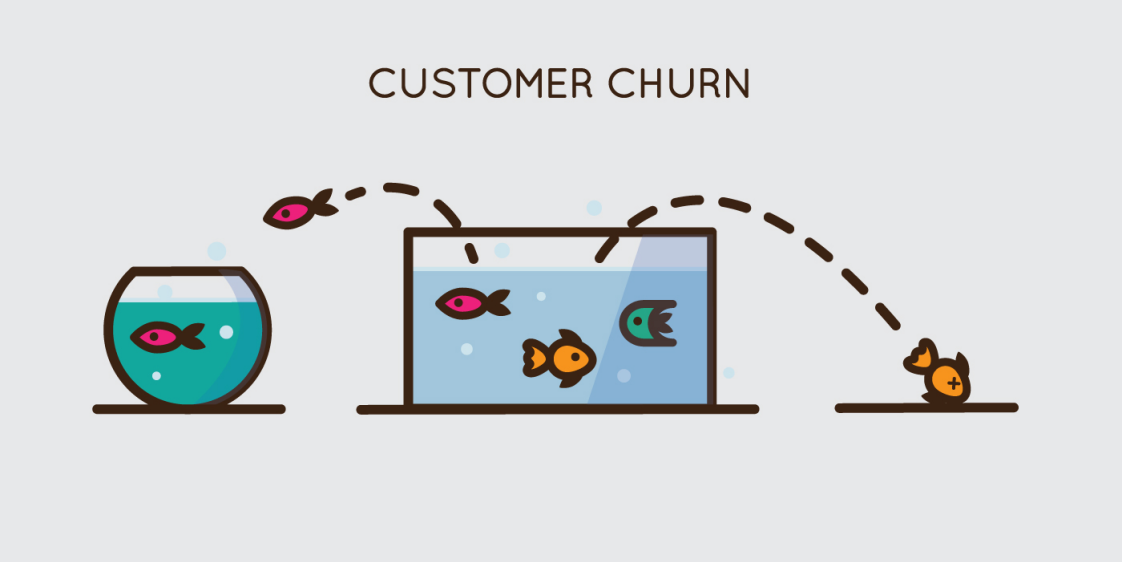
Author : Nassereddine BELGHITH, Data scientist at Lansrod Tech
Mail: nassereddine.belghith@lansrod.com

CHURN PREDICTION



Main items:

- Churn
- Classification
- Decision Tree
- Logistic regression
- Business-understanding
- Features-importance
- Cross-validation

- PySpark
- Python
- Plotly [1]

Introduction:

The churn :

The churn rate is one of the most significant KPIs to measure the performance of an online service whose business model is based on the growth of subscribers or users. It determines whether the proposed service is in line with their expectations.
In a way, the attrition rate measures customer **loyalty** and **satisfaction**, knowing that it is much more expensive to search for new ones than to retain existing ones.
In this project I aim to predict the churn and create a real customer retention program using Python as a programming language and Spark as a distributed data processing engine. For this, I think that I should answer the question:

Why using Spark and Python ?

Apache Spark is one of the most widely used frameworks to handle and work with **Big Data**, generally it is not a data scientist task, and Python is one of the most widely used programming languages for **Data Analysis** and **Machine Learning**. So, why not use them together? This is where Spark with Python also known as PySpark comes into the picture.
Spark is an open-source cluster-computing framework for real-time processing developed by the Apache Software Foundation. It provides an interface for programming entire clusters with implicit data parallelism and fault-tolerance.

- Speed: It is 100x faster than traditional large-scale data processing frameworks like Apache Storm, Flink ...
- Powerful Caching: Simple programming layer provides powerful caching[2] and disk persistence capabilities.
- Deployment: Can be deployed through Mesos, Hadoop via Yarn, or Spark's own cluster manager.

[1] if you want to run my script you have to change the user name and the key for plotly.

[2] Caching is one of spark actions on RDD. It is a mechanism to speed up applications that access the same RDD multiple times. An RDD that is not cached, nor checkpointed, is re-evaluated again each time an action is invoked on that RDD. There are two function calls for caching an RDD: cache() and persist(level: StorageLevel).

- Real Time: Real-time computation and low latency because of in-memory computation[3].
- Polyglot: It is one of the most important features of this framework as it can be programmed in Scala, Java, Python, and R.

For more information please visit the official Apache spark Documentation. (Here is the link).

Note:

In this project I have used python as a programming language.
I haven't used scikit-learn package for machine learning processing however I have used MLib (Apache Spark scalable Machine Learning Library).

Spark:

To more understand what spark is, here is a very simple architecture of this framework:

Input:

- RDD creation (Resilient Distributed DataSet)

||-From HDFS: RDD[Text], RDD[Text,Text]
||- -From Hbase:RDD[ImmutableBytesWritable,Result]
||- - -From Kafka: DStream[Text]
||- - - - From Elastic Search: RDD[string,Object]
||- - - - - From Hive : DF(RDD[Row])

Transformations:

- map
- filter
- reduce

Output:

- count
- write
- save as

[3]RDDs are stored in memory while being computed

Note: If you are new with Spark please note that there is a difference between the spark dataframe and the pandas dataframe.

Decision Tree :

When we speak about classification (one of the problems in machine learning) we look to the basic machine learning algorithms: Decision Tree, Logistic Regression, Random Forest and Naive Bayes. If you have a large amount of data you should use the random forest algorithm. The decision tree is actually a part of the random forest algorithm.
It is a tree shaped diagram to determine a course of action. Each branch of the tree represents a possible decision, occurrence or reaction.
Decision tree can be used for classification (True/false ,Yes/No, Male/female...) and regression when the variable is continuous.

Advantages of decision tree:

- It is simple to understand,interpret and visualize.
- Little effort required for data preparation.
- Non linear parameters don't affect its performance.

Disadvantages of decision tree:

- Overfitting: It occurs when the algorithm captures noise in the data.
- High variance: The model can get unstable due to small variation data.
- Low biased tree: A highly complicated Decision tree tends to have a low bias which makes it difficult for the model to work with new data.

Here is the decision tree algorithm [4]:

[4]There are some key words that you have to know; entropy and gain

```
INPUT: S, where S = set of classified instances
OUTPUT: Decision Tree
Require: S ≠ ∅, num_attributes > 0
 1: procedure BuildTree
 2:     repeat
 3:         maxGain ← 0
 4:         splitA ← null
 5:         e ← Entropy(Attributes)
 6:         for all Attributes a in S do
 7:             gain ← InformationGain(a, e)
 8:             if gain > maxGain then
 9:                 maxGain ← gain
10:                 splitA ← a
11:             end if
12:         end for
13:         Partition(S, splitA)
14:     until all partitions processed
15: end procedure
```

Data exploration and preparation :

- In this project data is stored on a csv file.
- Each row represents a customer, each column contains customer's attributes.
- The data set includes information about:

Churn: Customers who left within the last month.
Services: Service that each customer has signed up for – phone, multiple lines, internet, online security, online backup, device protection, tech support, and streaming TV and movies.
Customer account information: How long has he been a customer ? Contract, payment method, paperless billing, monthly charges, and total charges.
Demographic information about customers: gender, age range, and has he partners and dependants ?
You can find more information about used Data on Kaggle: Link to Data on Kaggle

- CustomerID: The Customer ID.
- Gender : Whether the customer is a male or a female.
- SeniorCitizen: Whether the customer is a senior citizen or not (1, 0).
- Partner: Whether the customer has a partner or not (Yes, No).
- Dependents : Whether the customer has dependants or not (Yes, No).

- Tenure: Number of months the customer has stayed with the company.
- PhoneService: Whether the customer has a phone service or not (Yes, No).
- MultipleLines: Whether the customer has multiple lines or not (Yes, No, No phone service).
- InternetService: Customer's internet service provider (DSL, Fiber optic, No).
- OnlineSecurity: Whether the customer has online security or not (Yes, No, No internet service).
- OnlineBackup : Whether the customer has online backup or not (Yes, No, No internet service).
- DeviceProtection: Whether the customer has device protection or not (Yes, No, No internet service).
- TechSupport: Whether the customer has tech support or not (Yes, No, No internet service).
- StreamingTV: Whether the customer has streaming TV or not (Yes, No, No internet service).
- StreamingMovies: Whether the customer has streaming movies or not (Yes, No, No internet service).
- Contract: The contract term of the customer (Month-to-month, One year, Two year).
- PaperlessBilling: Whether the customer has paperless billing or not (Yes, No).
- PaymentMethod: The customer's payment method (Electronic check, Mailed check, . . .).
- MonthlyCharges: The amount charged to the customer monthly.
- TotalCharges: The total amount charged to the customer.
- Churn: Whether the customer churned or not (Yes or No).

As the decision tree algorithm can not handle some types (String for example), I converted the data to numerical. Using the withColumn instance in Pyspark, I have manually achieved a one-hot-encoding.

The purpose of the study is to find why customers unsubscribe rather than building the best ML model to predict customer churn for this I have used only two algorithms (Decision Tree and Logistic Regression).
I have converted all the features to numerical then I have chosen the most suitable. I can then see the importance of each variable and keep only the relevant ones.

Data Exploration:

[33]:
```
pd.DataFrame(final_test_data.take(5), columns=final_test_data.columns).transpose()
```

[33]:

	0	1	2	3	4
gender	0.00	1.00	1.00	1.00	0.00
SeniorCitizen	0.00	0.00	0.00	0.00	0.00
Partner	1.00	0.00	0.00	0.00	0.00
Dependents	0.00	0.00	0.00	0.00	0.00
tenure	1.00	34.00	2.00	45.00	2.00
PhoneService	0.00	1.00	1.00	0.00	1.00
MultipleLines	0.00	0.00	0.00	0.00	0.00
OnlineSecurity	0.00	1.00	1.00	1.00	0.00
OnlineBackup	1.00	0.00	1.00	0.00	0.00
DeviceProtection	0.00	1.00	0.00	1.00	0.00
TechSupport	0.00	0.00	0.00	1.00	0.00
StreamingTV	0.00	0.00	0.00	0.00	0.00
StreamingMovies	0.00	0.00	0.00	0.00	0.00
PaperlessBilling	1.00	0.00	1.00	0.00	1.00
MonthlyCharges	29.85	56.95	53.85	42.30	70.70
TotalCharges	29.85	1889.50	108.15	1840.75	151.65
Churn	0.00	0.00	1.00	0.00	1.00
InternetService_DSL	1.00	1.00	1.00	1.00	0.00

Figure 1: Some Encoded features

[88]:
```
#display a summary of some column of our dataframe
display(train_new.select("Churn","MonthlyCharges","TotalCharges").describe().toPandas())
#Subscribers pay a lot ( 18 to 119 $ per month)
```

	summary	Churn	MonthlyCharges	TotalCharges
0	count	7043	7043	7032
1	mean	0.2653698707936959	64.76169246059922	2283.3004408418697
2	stddev	0.44156130512194697	30.090047097678482	2266.771361883145
3	min	0	18.25	18.8
4	max	1	118.75	8684.8

Note: Most of the customers are paying a really expensive subscription 18 to 119 $ per month.

[39]:

	0	1	2	3	4
summary	count	mean	stddev	min	max
Churn	2	0.5	0.7071067811865476	0	1
count(CASE WHEN (Churn = 1) THEN true END)	2	934.5	1321.5825740376574	0	1869

[41]:
```
print('Percentage of Unsubscribed clients =',(1869/7043*100),'%')
#26.5 % of clients have unscubscribed
```

```
Percentage of Unsubscribed clients = 26.536987079369588 %
```

Note: Only 26.5 % of the customers have unsubscribed.

[34]:

	0	1	2	3	4
summary	count	mean	stddev	min	max
InternetService_DSL	2	0.5	0.7071067811865476	0	1
count(CASE WHEN (Churn = 1) THEN true END)	2	934.5	672.4585489084067	459	1410
count(CASE WHEN (InternetService_DSL = 1) THEN true END)	2	1210.5	1711.9055172526316	0	2421

[39]:
```
print('Percentage of Unsubscribed clients who have DSL internet service =',(2421/7043*100),'%')
#34 % of unscubscribers have DSL internet service
```

```
Number of unscubscribers who have DSL internet service = 34.37455629703251 %
```

Note: Only 34% of them have the DSL and a device protection.

```
train_bd2 = train_new.groupBy("InternetService_Optic_Fiber").agg(count(when((col("InternetService_Optic_Fiber")==1),True)))
train_bd2.describe().toPandas().transpose()
```

	0	1	2	3	4
summary	count	mean	stddev	min	max
InternetService_Optic_Fiber	2	0.5	0.7071067811865476	0	1
count(CASE WHEN (InternetService_Optic_Fiber = 1) THEN true END)	2	1548.0	2189.202594553551	0	3096

```
print('Percentage of Unsubscribed clients who have optic fiber internet service',(3096/7043*100),'%')
#43.95 % of clients have optic fiber internet service
```

```
Percentage of Unsubscribed clients who have optic fiber internet service 43.958540394718156 %
```

Note: Almost 44% of the customers have the optical fiber internet service.
Paperless billing:

[46]:

	0	1	2	3	4
summary	count	mean	stddev	min	max
PaperlessBilling	2	0.5	0.7071067811865476	0	1
count(CASE WHEN (PaperlessBilling = 1) THEN true END)	2	2085.5	2949.34238432909	0	4171

[48]:
```
print('Percentage of Unsubscribed clients who use PaperlessBilling service',(4171/7043*100),'%')
#59.22 % of clients have optic fiber internet service
```

```
Percentage of Unsubscribed clients who use PaperlessBilling service 59.22192247621753 %
```

Note: 60% of the customers subscribe to the paperless billing service.

Payment method:
Note: Only 43% of the customers have an automatic payment method.

Phone service:

[52]:

	0	1	2	3	4
summary	count	mean	stddev	min	max
PhoneService	2	0.5	0.7071067811865476	0	1
count(CASE WHEN (PhoneService = 1) THEN true END)	2	3180.5	4497.906235127629	0	6361

[53]:
```
print('Percentage of clients who have phone service ',(6361/7043*100),'%')
# 50.47% of male clients
```

```
Percentage of clients who have phone service  90.31662643759762 %
```

Note: 90% of the customers have a phone service.

Gender:

[49]:

	0	1	2	3	4
summary	count	mean	stddev	min	max
gender	2	0.5	0.7071067811865476	0	1
count(CASE WHEN (gender = 1) THEN true END)	2	1777.5	2513.7646071181766	0	3555

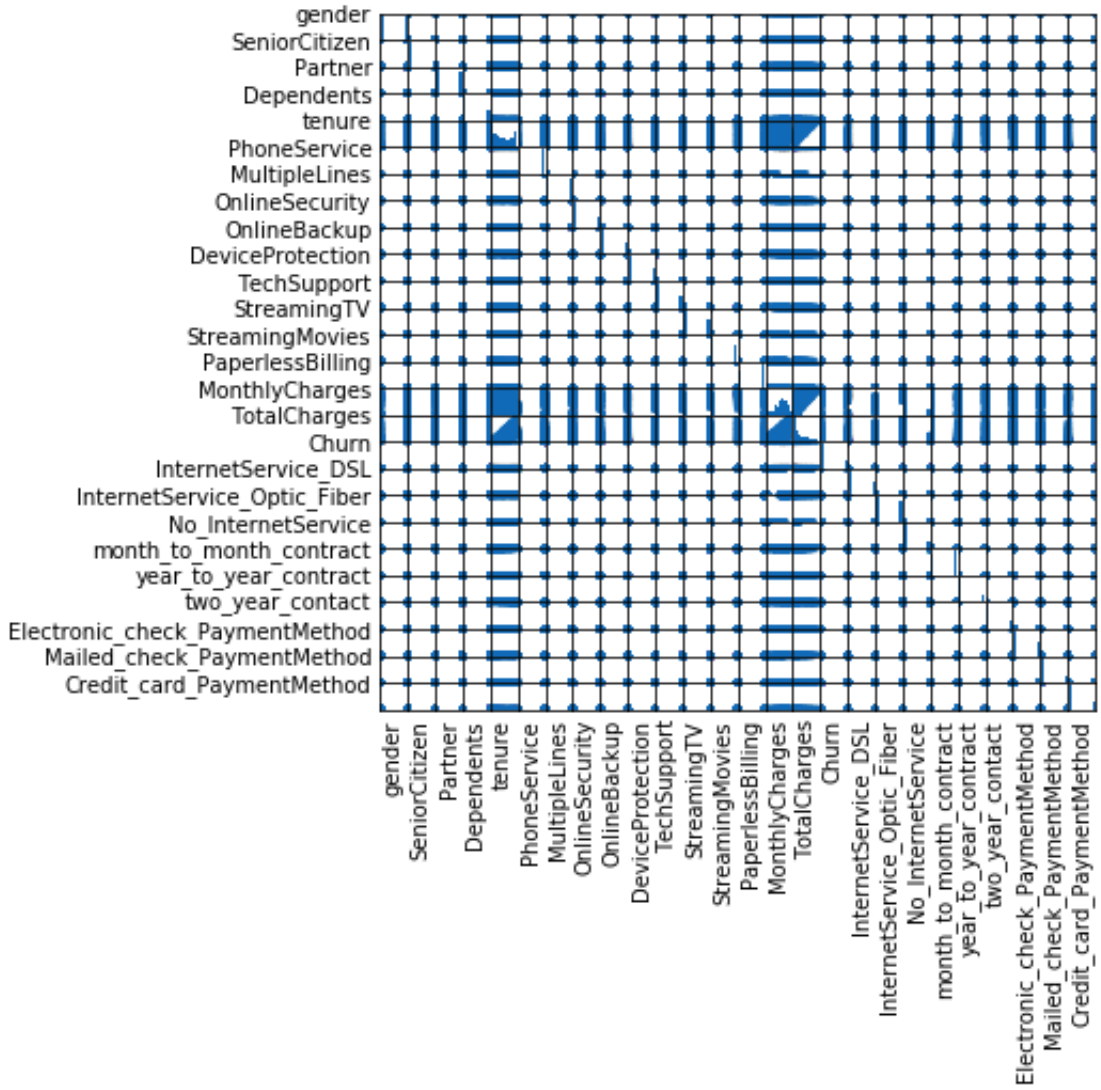
[51]:
```
print('Percentage of male clients',(3555/7043*100),'%')
# 50.47% of male clients
```

```
Percentage of male clients 50.4756495811444 %
```

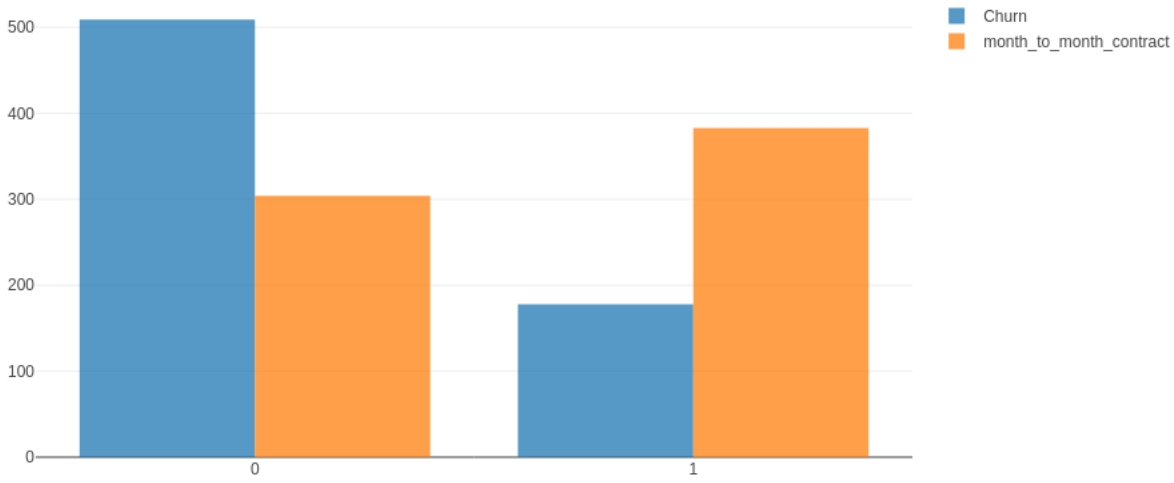
Note: The data contains as many men as women.
Contract:
Note: Most of the customers prefer a **month-to-month contract**.
We have a real panel of different customers: Some subscribed a long time ago and others are really new.
Churn:

```
[67]: sampled_data.corr()['Churn'].sort_values()
```

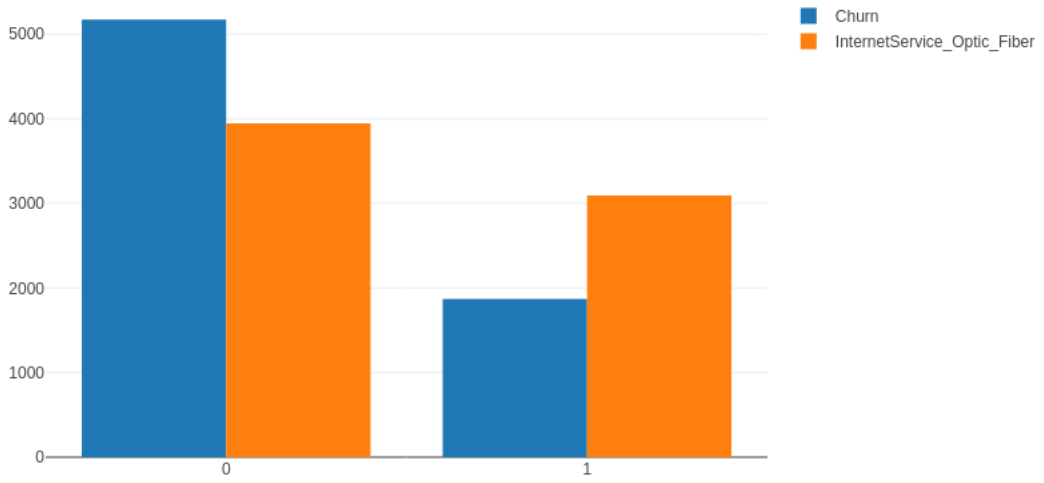
```
[67]: tenure                                -0.329324
      two_year_contract                     -0.270258
      year_to_year_contract                 -0.206863
      TotalCharges                          -0.180694
      No_InternetService                    -0.177126
      TechSupport                           -0.176730
      OnlineSecurity                        -0.174197
      Credit_card_PaymentMethod             -0.158046
      InternetService_DSL                   -0.152424
      Dependents                            -0.150585
      Partner                               -0.116037
      OnlineBackup                          -0.100158
      Mailed_check_PaymentMethod            -0.084489
      DeviceProtection                      -0.056976
      gender                                 0.000135
      StreamingTV                            0.029794
      StreamingMovies                        0.076932
      PhoneService                           0.082415
      SeniorCitizen                          0.106816
      MultipleLines                          0.116112
      PaperlessBilling                       0.142839
      MonthlyCharges                         0.181744
      Electronic_check_PaymentMethod         0.248390
      InternetService_Optic_Fiber            0.286975
      month_to_month_contract                0.399764
      Churn                                  1.000000
      Name: Churn, dtype: float64
```



The risk of churn is highly correlated to 3 features:
The tenure: Customers for a long period usually do not unsubscribe.
Customers having the optical fiber tend more to unsubscribe (Is there any problem with the quality of the optical fiber proposed by the telecommunication company?).
Customers having a month-to-month contract tend more to unsubscribe (The telecommunication company should propose promotional offers on the one-year and two-year contracts to avoid un-subscriptions)



The two histograms confirm the assumptions, we had when we looked at the correlation matrix.



The more customers sign up for optic fiber service the more they tend to unsubscribe.

Data Modeling, Features Importance and Model Evaluation:

By interpreting the correlation between the churn and features we can conclude that some features don't largely affect the the costumer choice for example : the gender (rational consumer either it is male or female), Streaming TV service (For this company the Streaming TV service is working well for that it doesn't affect the client choice) ...
I have dropped all features whose correlation to the churn is weak.

```
[52]: final_test_data = final_test_data.drop('PhoneService').drop('StreamingMovies').drop('gender')\
.drop('PaperlessBilling').drop('MonthlyCharges').drop('SeniorCitizen').drop('StreamingTV')\
.drop('TechSupport').drop('InternetService_DSL').drop('Credit_card_PaymentMethod')\
.drop('OnlineBackup').drop('Dependents').drop('year_to_year_contract').drop('Partner')\
.drop('OnlineSecurity').drop('TotalCharges').drop('MultipleLines').drop('StreamingTV')\
.drop('DeviceProtection').drop('Mailed_check_PaymentMethod').cache()
```

To work with decision tree and logistic regression we have to convert data into (label −−vector)

```
[61]: vectorized_CV_data.show()
```

```
+-----+--------------------+
|label|            features|
+-----+--------------------+
|    0|[2.0,1.0,0.0,0.0,...|
|    0|[45.0,0.0,0.0,0.0...|
|    1|[2.0,1.0,1.0,0.0,...|
|    1|[8.0,1.0,1.0,0.0,...|
|    0|[22.0,0.0,1.0,0.0...|
|    1|[28.0,1.0,1.0,0.0...|
|    0|[13.0,0.0,0.0,0.0...|
|    0|[49.0,1.0,1.0,0.0...|
|    0|[10.0,1.0,0.0,0.0...|
|    1|[1.0,1.0,0.0,0.0,...|
|    0|[1.0,1.0,0.0,1.0,...|
|    0|[58.0,0.0,0.0,0.0...|
|    1|[47.0,1.0,1.0,0.0...|
|    1|[1.0,1.0,0.0,0.0,...|
|    0|[17.0,1.0,0.0,0.0...|
|    0|[1.0,0.0,0.0,0.0,...|
|    1|[5.0,1.0,1.0,0.0,...|
|    0|[46.0,0.0,1.0,0.0...|
|    1|[34.0,1.0,1.0,0.0...|
|    0|[11.0,1.0,1.0,0.0...|
+-----+--------------------+
only showing top 20 rows
```

```
[64]: vectorized_CV_data.printSchema()

root
 |-- label: long (nullable = true)
 |-- features: vector (nullable = true)
```

As you see I didn't give a huge importance to the type of the label (Long Type) because of the fact that I am working with a small dataset from kaggle as I mentioned below.

Cross Validation:

The decision tree accuracy:

```
f1 accuracy: 0.7074596937842901
```

]:

	indexedLabel	prediction	probability
0	0.0	0.0	[0.5077238605250115, 0.4922613947498844]
1	0.0	0.0	[0.7984249249580824, 0.20157507504191763]
2	1.0	1.0	[0.3147447012325525, 0.6852552987674475]
3	1.0	1.0	[0.3227340568195586, 0.6772659431804414]
4	0.0	1.0	[0.48405086485933063, 0.5159491351406693]

The logistic regression accuracy:

```
f1 accuracy: 0.721362518813804
```

	indexedLabel	prediction	probability
0	1.0	0.0	[0.6181818181818182, 0.38181818181818183]
1	0.0	0.0	[0.7838983050847458, 0.21610169491525424]
2	0.0	0.0	[0.5167095115681234, 0.4832904884318766]
3	0.0	0.0	[0.7838983050847458, 0.21610169491525424]
4	1.0	1.0	[0.28378378378378377, 0.7162162162162162]

Conclusion

- The optic fiber service has to be checked.
- It is better to engage costumer with one / two years contract.
- New customers are more able to unsubscribe it is better to present them special offers.
- The Fiber Optic is a bad option, most of the customers who subscribed to it cancel their subscription 'Month-to-month' contracts leave to much flexibility to the customers who can cancel their subscription anytime.
- The subscription must be simple to avoid massive unsubscriptions (it must do not have many options: streaming movies or tv for example).

- Automatic payment must be proposed as first payment solution in order to decrease the risk of churn.

Note: If you want to run my script please ask me to give you access.