Project Name: Interpreting-ML with Eli5

Table of Contents

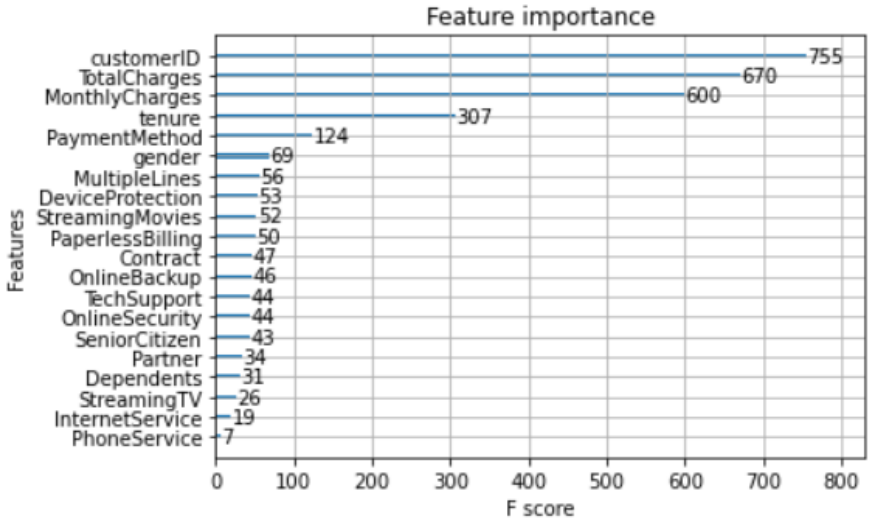
Demo

2.3 Feature Importance with Importance Type = 'Weight':

Importance Type = 'Wieght' is based on the number of times a feature appears i

```
In [10]: plot_importance(model,importance_type='weight')
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x23474a024c0>
```



In [14]:
```
# fit model on all training data
model = XGBClassifier()
model.fit(X_train, y_train)
```

Out[14]:
```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=0, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

4.1 Feature Importance through ELI5

In [17]:
```
eli5.show_weights(model.get_booster())
```

Out[17]:

Weight	Feature
0.4379	Contract
0.0752	OnlineSecurity
0.0734	InternetService
0.0535	TechSupport
0.0329	tenure
0.0310	PaperlessBilling
0.0282	MultipleLines
0.0250	StreamingMovies
0.0237	MonthlyCharges
0.0227	Dependents
0.0224	PhoneService
0.0221	OnlineBackup
0.0219	TotalCharges
0.0203	PaymentMethod
0.0201	customerID
0.0199	SeniorCitizen
0.0190	gender
0.0183	DeviceProtection
0.0181	StreamingTV
0.0143	Partner

In [18]:
```
print(y_train.iloc[1])
show_prediction(model, X_test.iloc[1], feature_names = X.columns.tolist(),
                show_feature_values=True)
```

```
Churn    0
Name: 1671, dtype: int32
```

Out[18]: **y=0** (probability **0.961**, score **-3.206**) top features

Contribution[?]	Feature	Value
+1.288	<BIAS>	1.000
+0.775	Contract	1.000
+0.690	tenure	64.000
+0.516	customerID	3713.000
+0.295	OnlineSecurity	2.000
+0.267	Partner	0.000
+0.216	MultipleLines	0.000
+0.111	OnlineBackup	2.000
+0.057	gender	1.000
+0.046	SeniorCitizen	0.000
+0.016	PhoneService	1.000
+0.012	PaymentMethod	1.000
-0.004	Dependents	0.000
-0.028	InternetService	1.000
-0.040	PaperlessBilling	1.000
-0.068	StreamingTV	2.000
-0.071	TechSupport	0.000
-0.234	MonthlyCharges	1287.000
-0.271	StreamingMovies	2.000
-0.368	TotalCharges	5271.000

Overview

This is diving into Interpretation of ML Models using Eli5 Concept.
Interpretability is a relation between formal theories that expresses the possibility of interpreting or translating one into the other.
This repository contains the code for Interpretation of ML Models using python's various libraries.
It used Numpy, Pandas, Matplotlib, Seaborn, sklearn and eli5 libraries.
These libraries help to perform individually one particular functionality.
Numpy is used for working with arrays. It stands for Numerical Python.
Pandas objects rely heavily on Numpy objects.
Sklearn has 100 to 200 models.
Matplotlib is a plotting library.
Seaborn is data visualization library based on matplotlib.
ELI5 is a Python package which helps to debug machine learning classifiers and explain their predictions.
String module contains a number of functions to process standard Python strings.
The purpose of creating this repository is to gain insights into Interpretation of ML Models.
These python libraries raised knowledge in discovering these libraries with practical use of it.
It leads to growth in my ML repository.
This above few screenshots will help you to understand flow of output.

Motivation

The reason behind building this is, because interpretability is critical for data scientists, researchers and developers to explain their models and understand the value and accuracy of their findings. Interpretability is also important to debug machine learning models and make informed decisions about how to improve them as those decisions have an impact on worth and brand value majorly.

Technical Aspect

Numpy contains a multi-dimensional array and matrix data structures. It works with the numerical data. Numpy is faster because is densely packed in memory due to its homogeneous type. It also frees the memory faster.
Pandas module mainly works with the tabular data. It contains DataFrame and Series. Pandas is 18 to 20 times slower than Numpy. Pandas is seriously a game changer when it comes to cleaning, transforming, manipulating and analyzing data.
Matplotlib is used for EDA. Visualization of graphs helps to understand data in better way than numbers in table format. Matplotlib is mainly deployed for basic plotting. It consists of bars, pies, lines, scatter plots and so on. Inline command display visualization inline within frontends like in Jupyter Notebook, directly below the code cell that produced it.
Seaborn provides a high-level interface for drawing attractive and informative statistical graphics. It provides a variety of visualization patterns and visualize random distributions.

Sklearn is known as scikit learn. It provides many ML libraries and algorithms for it. It provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.

ELI5 means Explain Like I am 5.

There are two main ways to look at a classification or a regression model:

1. inspect model parameters and try to figure out how the model works globally;
2. inspect an individual prediction of a model, try to figure out why the model makes the decision it makes.

For (1) ELI5 provides `eli5.show_weights()` function;

For (2) it provides `eli5.show_prediction()` function.

ELI5 aims to handle not only simple cases, but even for simple cases having a unified API for inspection has a value:

1)you can call a ready-made function from ELI5 and get a nicely formatted result immediately;

2)formatting code can be reused between machine learning frameworks;

3)'drill down' code like feature filtering or text highlighting can be reused;

Need to train_test_split - Using the same dataset for both training and testing leaves room for miscalculations, thus increases the chances of inaccurate predictions.

The `train_test_split` function allows you to break a dataset with ease while pursuing an ideal model. Also, keep in mind that your model should not be overfitting or underfitting.

Accuracy_score : Refer to following link.

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

Sklearn provides a very efficient tool for encoding the levels of categorical features into numeric values.

LabelEncoder encode labels with a value between 0 and n_classes-1 where n is the number of distinct labels. If a label repeats it assigns the same value to as assigned earlier. Label Encoding refers to converting the labels into numeric form so as to convert it into the machine-readable form. Machine learning algorithms can then decide in a better way on how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

XGBoost is an implementation of gradient boosted decision trees designed for speed and performance that is dominative competitive machine learning.

Installation

Using intel core i5 9th generation with NVIDIA GFORECE GTX1650.

Windows 10 Environment Used.

Already Installed Anaconda Navigator for Python 3.x

The Code is written in Python 3.8.

If you don't have Python installed then please install Anaconda Navigator from its official site.

If you are using a lower version of Python you can upgrade using the pip package, ensuring you have the latest version of pip, *python* -m pip install --*upgrade pip and press Enter*.

Run/How to Use/Steps

Keep your internet connection on while running or accessing files and throughout too.
Follow this when you want to perform from scratch.

Open Anaconda Prompt, Perform the following steps:

cd <PATH>
pip install numpy
pip install pandas
pip install matplotlib
pip install seaborn
pip install sklearn
pip install eli5

You can also create requirement.txt file as, pip freeze > requirements.txt

run files.

Follow this when you want to just perform on local machine.
Download ZIP File.
Right-Click on ZIP file in download section and select Extract file option, which will unzip file.
Move unzip folder to desired folder/location be it D drive or desktop etc.
Open Anaconda Prompt, write cd <PATH> and press Enter.
eg: cd C:\Users\Monica\Desktop\Projects\Python Projects 1\ allin1\Project_6_InterpretingML

In Anconda Prompt, pip install -r requirements.txt to install all packages.
Open in Jupyter Notebook, <filename>.ipynb
That is,

Open in Jupyter Notebook, 1)InterpretingML_with_Eli5.ipynb
This takes WA_Fn-UseC_-Telco-Customer-Churn_dataset.csv file as input dataset.
Please be careful with spellings or numbers while typing filename and easier is just copy filename and then run it to avoid any silly errors.
Note: cd <PATH>
[Go to Folder where file is. Select the path from top and right-click and select copy option and paste it next to cd one space <path> and press enter, then you can access all files of that folder]
[cd means change directory]

Directory Tree/Structure of Project

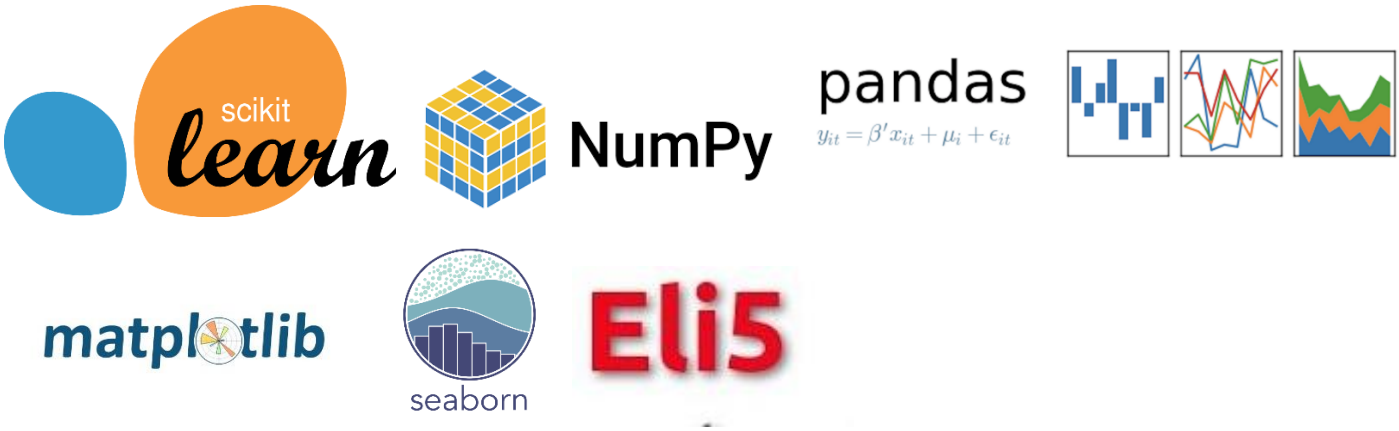
Folder: allin1 > Project_6_InterpretingML

1)InterpretingML_with_Eli5.ipynb

To Do/Future Scope

Can try another complex dataset with same library.

Technologies Used/System Requirements/Tech Stack



Credits

Pradeep Sir