learn-co-curriculum / dsc-fitting-a-logistic-regression-model-lab Public

View license

0 stars 167 forks

Star Watch

Code Issues Pull requests Actions Projects Security Insights

solution

This branch is 10 commits ahead, 11 commits behind master.

sumedh10 update readme ... on Nov 21, 2019 11

View code

README.md

Fitting a Logistic Regression Model - Lab

Introduction

In the last lesson you were given a broad overview of logistic regression. This included an introduction to two separate packages for creating logistic regression models. In this lab, you'll be investigating fitting logistic regressions with `statsmodels`. For your first foray into logistic regression, you are going to attempt to build a model that classifies whether an individual survived the Titanic shipwreck or not (yes, it's a bit morbid).

Objectives

In this lab you will:

- Implement logistic regression with `statsmodels`
- Interpret the statistical results associated with model parameters

Import the data

Import the data stored in the file `'titanic.csv'` and print the first five rows of the DataFrame to check its contents.

```
# Import the data
import pandas as pd

df = pd.read_csv('titanic.csv')
df.head()
```

<style scoped> .dataframe tbody tr th:only-of-type { vertical-align: middle; }

```
.dataframe tbody tr th {
    vertical-align: top;
}

.dataframe thead th {
    text-align: right;
}
```

</style>

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Pa
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0
3	4	1	1	Futrelle, Mrs. Jacques Heath	female	35.0	1	0

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Pa
				(Lily May Peel)				
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0

Define independent and target variables

Your target variable is in the column `'Survived'` . A `0` indicates that the passenger didn't survive the shipwreck. Print the total number of people who didn't survive the shipwreck. How many people survived?

```
# Total number of people who survived/didn't survive
df['Survived'].value_counts()
```

```
0    549
1    342
Name: Survived, dtype: int64
```

Only consider the columns specified in `relevant_columns` when building your model. The next step is to create dummy variables from categorical variables. Remember to drop the first level for each categorical column and make sure all the values are of type `float` :

```
# Create dummy variables
relevant_columns = ['Pclass', 'Age', 'SibSp', 'Fare', 'Sex', 'Embarked', 'Survived']
dummy_dataframe = pd.get_dummies(df[relevant_columns], drop_first=True, dtype=float)

dummy_dataframe.shape
```

```
(891, 8)
```

Did you notice above that the DataFrame contains missing values? To keep things simple, simply delete all rows with missing values.

> NOTE: You can use the `.dropna()` method to do this.

```
# Drop missing rows
dummy_dataframe = dummy_dataframe.dropna()
dummy_dataframe.shape
```

```
(714, 8)
```

Finally, assign the independent variables to `X` and the target variable to `y`:

```
# Split the data into X and y
y = dummy_dataframe['Survived']
X = dummy_dataframe.drop(columns=['Survived'], axis=1)
```

Fit the model

Now with everything in place, you can build a logistic regression model using `statsmodels` (make sure you create an intercept term as we showed in the previous lesson).

> Warning: Did you receive an error of the form "LinAlgError: Singular matrix"? This means that `statsmodels` was unable to fit the model due to certain linear algebra computational problems. Specifically, the matrix was not invertible due to not being full rank. In other words, there was a lot of redundant, superfluous data. Try removing some features from the model and running it again.

```
# Build a logistic regression model using statsmodels
import statsmodels.api as sm
X = sm.tools.add_constant(X)
logit_model = sm.Logit(y, X)
result = logit_model.fit()
```

```
Optimization terminated successfully.
         Current function value: 0.443267
         Iterations 6
```

```
//anaconda3/lib/python3.7/site-packages/numpy/core/fromnumeric.py:2389:
FutureWarning: Method .ptp is deprecated and will be removed in a future version.
Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Analyze results

Generate the summary table for your model. Then, comment on the p-values associated with the various features you chose.

```
# Summary table
result.summary()
```

Logit Regression Results

Dep. Variable:	Survived	**No. Observations:**	714
Model:	Logit	**Df Residuals:**	706
Method:	MLE	**Df Model:**	7
Date:	Wed, 20 Nov 2019	**Pseudo R-squ.:**	0.3437
Time:	16:18:35	**Log-Likelihood:**	-316.49
converged:	True	**LL-Null:**	-482.26
Covariance Type:	nonrobust	**LLR p-value:**	1.103e-67

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.6503	0.633	8.921	0.000	4.409	6.892
Pclass	-1.2118	0.163	-7.433	0.000	-1.531	-0.892
Age	-0.0431	0.008	-5.250	0.000	-0.059	-0.027
SibSp	-0.3806	0.125	-3.048	0.002	-0.625	-0.136
Fare	0.0012	0.002	0.474	0.636	-0.004	0.006
Sex_male	-2.6236	0.217	-12.081	0.000	-3.049	-2.198
Embarked_Q	-0.8260	0.598	-1.381	0.167	-1.999	0.347
Embarked_S	-0.4130	0.269	-1.533	0.125	-0.941	0.115

```
# Based on our P-values, most of the current features appear to be significant based
# That said, the 'Embarked' and 'Fare' features were not significant based on their
```

Level up (Optional)

Create a new model, this time only using those features you determined were influential based on your analysis of the results above. How does this model perform?

```
# Your code here
relevant_columns = ['Pclass', 'Age', 'SibSp', 'Sex', 'Survived']
dummy_dataframe = pd.get_dummies(df[relevant_columns], drop_first=True, dtype=float)

dummy_dataframe = dummy_dataframe.dropna()

y = dummy_dataframe['Survived']
X = dummy_dataframe.drop(columns=['Survived'], axis=1)

X = sm.tools.add_constant(X)
logit_model = sm.Logit(y, X)
result = logit_model.fit()

result.summary()
```

```
Optimization terminated successfully.
         Current function value: 0.445882
         Iterations 6
```

Logit Regression Results

Dep. Variable:	Survived	**No. Observations:**	714
Model:	Logit	**Df Residuals:**	709
Method:	MLE	**Df Model:**	4
Date:	Wed, 20 Nov 2019	**Pseudo R-squ.:**	0.3399
Time:	16:18:50	**Log-Likelihood:**	-318.36
converged:	True	**LL-Null:**	-482.26
Covariance Type:	nonrobust	**LLR p-value:**	1.089e-69

	coef	std err	z	P>|z|	[0.025	0.975]
const	5.6008	0.543	10.306	0.000	4.536	6.666
Pclass	-1.3174	0.141	-9.350	0.000	-1.594	-1.041

Age	-0.0444	0.008	-5.442	0.000	-0.060	-0.028
SibSp	-0.3761	0.121	-3.106	0.002	-0.613	-0.139
Sex_male	-2.6235	0.215	-12.229	0.000	-3.044	-2.203

```
# Comments:
# Note how removing the insignificant features had little impact on the $R^2$ value
# of our model.
```

Summary

Well done! In this lab, you practiced using `statsmodels` to build a logistic regression model. You then interpreted the results, building upon your previous stats knowledge, similar to linear regression. Continue on to take a look at building logistic regression models in Scikit-learn!

Releases

No releases published

Packages

No packages published

Contributors 5

Languages

- Jupyter Notebook 100.0%