Eurovision Test Program

September 21, 2021

0.1 Predicting Scoring Patterns based on Song Order: Eurovision 2021

[1]:
```
import numpy as np
import pandas as pd
```

If we used Python to read the html_file, we would use the following code:

```
from pandas import read_html
test = pd.read_html("database/Eurovision Song Contest Database 2021.html")[1]
```

We would take the second element `[1]` here, because `[0]` is the initial text outside of the table.

0.1.1 Importing the Data

Instead of using `read_html` on the html file, we are going to take the newly formed csv file made in R / SQL, and make that the `test` database:

[2]:
```
# The year we are analyzing
year = 2021

# Instead, we are going to simply read the .csv created using R and SQL:
test = pd.read_csv("csv database/Eurovision Final %s Results.csv" % year)

# Drop the duplicate index column from the .csv file:
test = test.drop("Unnamed: 0", axis=1)
test.head()
```

[2]:

	Place	Points	Percent_Earned	Song_Order	Country
0	16	94	0.100427	1	Cyprus
1	21	57	0.060897	2	Albania
2	17	93	0.099359	3	Israel
3	19	74	0.079060	4	Belgium
4	9	204	0.217949	5	Russia

0.1.2 Visualizing the Data

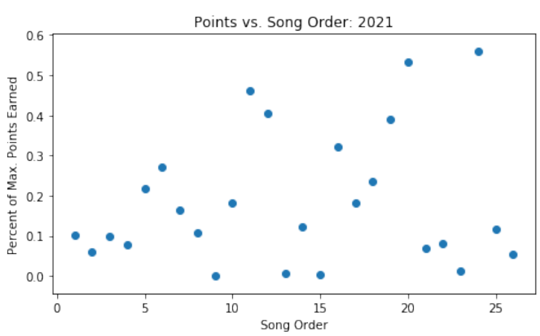
Now, we can import the following packages to help visualize the data and its' regression models:

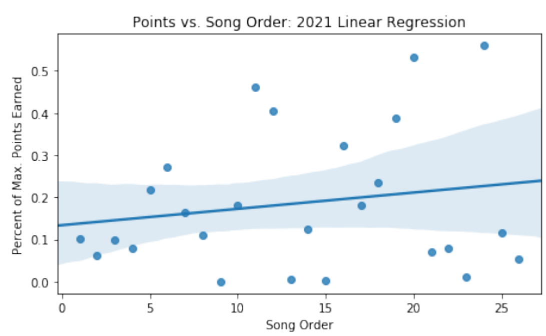
[3]:

```python
# Jupyter Notebook is weird, and requires this code to show all graphs when␣
↪running the entire code at once.
%matplotlib inline

# Now, let's import matplotlib and seaborn for visualizations. seaborn␣
↪specializes in regression.
import matplotlib.pyplot as plt
import seaborn as sb

# We need the following package to summarize the linear regression:
from statsmodels.formula.api import ols
```

Let's take the percent of points earned (Percent_Earned), and fit it based on the order of songs (Song_Order). We will view the graph without and with the linear regression line side-by-side:

[4]:
```python
points_lm = ols('Percent_Earned ~ Song_Order', data=test).fit()

# Plotting the scatter plot without and with the regression line side-by-side:
fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(16,4))

ax1.scatter(x='Song_Order', y='Percent_Earned', data=test)
ax1.set_xlabel("Song Order")
ax1.set_ylabel("Percent of Max. Points Earned")
ax1.set_title("Points vs. Song Order: %s" % year)

sb.regplot(x='Song_Order', y='Percent_Earned', data=test)
ax2.set_xlabel("Song Order")
ax2.set_ylabel("Percent of Max. Points Earned")
ax2.set_title("Points vs. Song Order: %s Linear Regression" % year)

plt.show()
```





There does not appear to be a significant correlation, according to the second graph and its linear regression formula. Let's confirm that hypothesis by fitting the data to an Ordinary Least Squares (OLS) regression model:

[5]:
```
points_lm = ols('Percent_Earned ~ Song_Order', data=test).fit()

print(points_lm.summary())
```

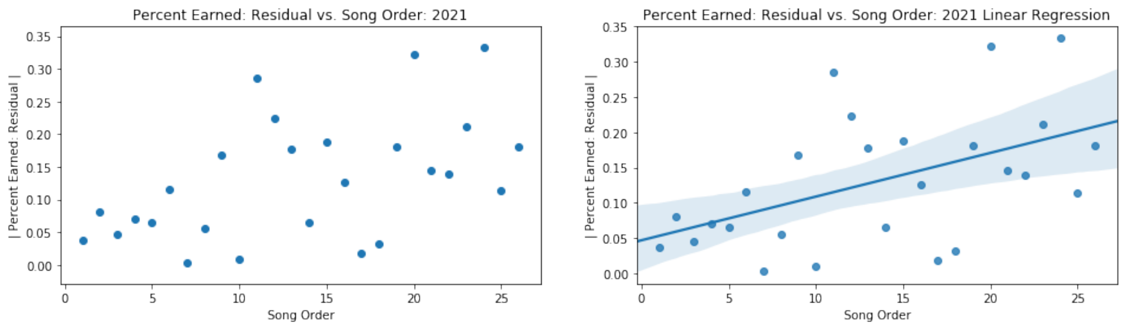
```
                            OLS Regression Results
==============================================================================
Dep. Variable:         Percent_Earned   R-squared:                       0.032
Model:                            OLS   Adj. R-squared:                 -0.008
Method:                 Least Squares   F-statistic:                    0.7938
Date:                Tue, 21 Sep 2021   Prob (F-statistic):              0.382
Time:                        17:26:36   Log-Likelihood:                 10.818
No. Observations:                  26   AIC:                            -17.64
Df Residuals:                      24   BIC:                            -15.12
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.1336      0.067      1.992      0.058      -0.005       0.272
Song_Order     0.0039      0.004      0.891      0.382      -0.005       0.013
==============================================================================
Omnibus:                        2.815   Durbin-Watson:                   1.876
Prob(Omnibus):                  0.245   Jarque-Bera (JB):                2.457
Skew:                           0.697   Prob(JB):                        0.293
Kurtosis:                       2.428   Cond. No.                         31.9
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

With a p-value of 0.382, it appears that song order and point percentage hardly have any linear correlation. Even with the `Song_Order` coefficient, there is only a slight positive correlation of `0.0039`. It seems like the song order does not sway the percent of points earned in either direction.

However, if we look at the graph again, we can tell that there's a lot more divergence in the points as song order increases. In other words, the points earned get less and less predictable as a song is placed later in the order. This time, let's take residuals from each point, and compare their magnitude (absolute value) to their respective song order:

[6]:
```
# Magnitude of Residuals from Percent_Earned:
test['Res_Abs'] = abs(points_lm.resid)

fig, (ax1, ax2) = plt.subplots(ncols=2, figsize=(16,4))

ax1.scatter(x='Song_Order', y='Res_Abs', data=test)
ax1.set_xlabel("Song Order")
ax1.set_ylabel("| Percent Earned: Residual |")
ax1.set_title("Percent Earned: Residual vs. Song Order: %s" % year)
```

```
sb.regplot(x='Song_Order', y='Res_Abs', data=test)
ax2.set_xlabel("Song Order")
ax2.set_ylabel("| Percent Earned: Residual |")
ax2.set_title("Percent Earned: Residual vs. Song Order: %s Linear Regression" % year)

plt.show()
```



The residuals appear to have a more direct linear trend with the song order. Based on the graph on the right, it appears that as songs go later in the order, their predictability becomes less accurate. This could imply that voters are more sporadic with how they vote as they listen to more songs.

Let's make sure that this prediction is significant within the data:

[7]:
```
points_res_lm = ols('Res_Abs ~ Song_Order', data=test).fit()

print(points_res_lm.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                Res_Abs   R-squared:                       0.256
Model:                            OLS   Adj. R-squared:                  0.225
Method:                 Least Squares   F-statistic:                     8.249
Date:                Tue, 21 Sep 2021   Prob (F-statistic):            0.00839
Time:                        17:26:37   Log-Likelihood:                 29.008
No. Observations:                  26   AIC:                            -54.02
Df Residuals:                      24   BIC:                            -51.50
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      0.0468      0.033      1.405      0.173      -0.022       0.116
Song_Order     0.0062      0.002      2.872      0.008       0.002       0.011
==============================================================================
```

```
Omnibus:                        1.001   Durbin-Watson:                   2.026
Prob(Omnibus):                  0.606   Jarque-Bera (JB):                0.887
Skew:                           0.412   Prob(JB):                        0.642
Kurtosis:                       2.626   Cond. No.                         31.9
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

It looks like we have a trend! Since the song order's p-value (0.008) is less than 0.01, we can say with 99% confidence that song order positively correlates with how points are distributed. This proves that later songs have a more volatile chance at earning higher and lower results, compared to their earlier counterparts.

[8]:
```
test['Fitted'] = points_lm.fittedvalues

test[['Place','Percent_Earned','Fitted','Res_Abs']].iloc[8:10+1,:]
```

[8]:

	Place	Percent_Earned	Fitted	Res_Abs
8	26	0.000000	0.168481	0.168481
9	10	0.181624	0.172352	0.009272
10	3	0.461538	0.176222	0.285317

[ ]:
```
```