**BA 355: Business Analytics, Case Study 2.2 (25 points)**

In this case study, we will use the credit score data from Case 2.1 to study how well Excel fits functions to data. Excel usually finds a *good* fit, but not necessarily the *best* fit. Use the data from the Case 2.2 Data file for exact results.

1) Calculate the loan default percentage for each credit score and then graph the percentages versus the credit scores. Adjust the x-axis so it starts at 300 instead of 0. Fit Excel’s automatic Exponential curve to the data and determine the equation and the r2 value. Include the graph here. It looks like a pretty good fit, no? You’ll need the coefficients and r2 for part 5) below so make a note of them.

**y = 22.848e-0.01x  
R² = 0.9834 (though when I manually write out the exp function and plug in these coefficients and use rsq(), I get about 0.71) The reason for this is discussed in the last question**

2) Now, use Solver to find the best fit. In other words, find the coefficients *a* and *b* for the equation that minimize the sum of the errors squared (unweighted). Calculate the r2 value too. Are these the same values you got in part 1)?

**Without constraints, A = 51,409.17 and B = -.0309, which seems clearly wrong**

**With constraints of a <= 100 and a >= 0, and b <= 0 and b >= -.1, I got a = 100 and b = -.091, which also is wrong**

**Finally, with the above constraints and multi-start bound variables, I get a = 11.33, b = -.0087.**

3) Now graph the data points versus this curve, making the curve look like a dashed line (so you can compare to the graph from part 1). Which of the two curves looks like a better “fit?”

**The custom line with the solver-derived values fits the data much more closely than the stock exponential function.**

4) Repeat what you did in part 2) but now use the weighted sum of the errors squared, factoring in the number of loans for each credit score.

**With the weighted sum of errors squared, I get a = 15.06 and b = -0.094.**

5) Fill in the table below with the coefficients and r2 values for each part:

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
|  | *a* | *b* | *r2* |
| 1) | 22.85 | -0.01 | 0.9834 |
| 2) | 11.33 | -0.0087 | 0.7913 |
| 3) | 15.06 | -0.0094 | 0.7677 |

We have now fit an exponential curve three different ways. Which one do you think is best, and why?

**I think that even though the R^2 value went down, the last combination of coefficients represents the data the most accurately because it is taking into effect sample size and sampling error by running solver under several different simulations.**

6) How does Excel come up with the coefficients *a* and *b* in part 1)? Excel does not use Solver for this. Figure this out on your own or do a deep Google search, then explain it to me. **DO NOT SHARE this result with other students,** this question is supposed to separate the A’s from the A+’s; however, you are welcome, even encouraged, to compare your answers from 1) through 5).3

SKIP FOLLOWING TEXT TO AVOID OVERVIEW OF SOLVER

The way that solver works depends on the data. For linear data, solver uses linear programming methods and finds a globally optimal solution. This is because all linear functions are convex, and convex optimization problems have globally optimal solutions. However, for concave (more formally nonconvex) data and non-smooth data, solver finds a locally optimal solution using GRG nonlinear solving methods.

When solver finds a local solution to a minimization problem, it means that the Karush-Kuhn-Tucker conditions have been satisfied by a valley, however this is not guaranteed to be the minimum. Indeed, when solver converges to a solution, it means that the absolute value of the relative chance in the objective function is less than the user defined value in the convergence box for the past five iterations (default is 0.0001).

If solver does this using Newton’s method, then it finds the root of an affine (matrix of linear transformation and translations) approximation of the function

START HERE TO SKIP SOLVER DISCUSSION

HOWEVER, excel does not use solver to fit an exponential trendline. Instead, it log transforms the original y values to determine fitted values, so it is not available for non-positive data. It does not use sum of squared deviations between the data and the prediction the way that solver does. Rather, it takes the log of both sides of exp formula

y=a\*e^bx

such that

ln(y)=ln(a) + bx

and

ln(y) = b+mx

where y is the dependent variable and x is the independent variable in a linear regression and the above equation is used to minimize the sum of squared deviations. This is why the R squared is inaccurate, because it transforms the data and then minimizes errors instead of minimizing the errors between the prediction and data.

Sources:

<https://www.mikemiddleton.com/Excel-Exponential-Curve-Fit-2010.pdf>

<https://www.ablebits.com/office-addins-blog/excel-trendline-types-equations-formulas/>

<https://www.solver.com/excel-solver-grg-nonlinear-solving-method-stopping-conditions>