Math 3660 - Spring 2022
Mathematical Models in Economics

Steven Tschantz

1/17/22

Profit maximizing firms

- Assignment 3

Instructions

Save a copy of this notebook, complete the exercises, save and submit on Brightspace your final version by start of class Tues. Feb. 8.

Nikhil Jayswal

Edit the line above that reads Enter your name here, inserting instead your name. You may also want to save to a file name that includes your name. By adding your name into the file and file name, you reduce the chance I will mix-up your submission with someone else's.

Exercises

1. [Calibration] Suppose consumers are choosing whether or not to buy a single product. Suppose their values for the product are normally distributed, that 20% of consumers buy the product at a price of $25, and they have an elasticity of demand of -2 at this price.

This first exercise is a calibration problem. Assuming a model and some basic information, determine coefficients of the model fitting the given information.

Use the method of undetermined coefficients assuming the distribution of consumer values is

In[1]:=
```
Clear[dist, mu, sigma, choiceprob]
```

In[2]:=
```
dist = NormalDistribution[mu, sigma]
```

Out[2]=
```
NormalDistribution[mu, sigma]
```

and the choice probability at a price *p* implied by these values is given by

In[3]:=
```
choiceprob[p_] = 1 - CDF[dist, p]
```

Out[3]=
$$1 - \frac{1}{2}\,\text{Erfc}\left[\frac{\text{mu} - \text{p}}{\sqrt{2}\ \text{sigma}}\right]$$

Write down a definition of the elasticity of this demand (probability) ...

The elasticity of this demand is the fractional change in choice probability, i.e. the probability that the consumer chooses the product, for some fractional change in price of the product.

In[4]:=
```
elasticity[p_] = D[choiceprob[p], p] * (p / choiceprob[p])
```

Out[4]=
$$-\frac{e^{-\frac{(\text{mu}-\text{p})^2}{2\,\text{sigma}^2}}\,\text{p}}{\sqrt{2\,\pi}\ \text{sigma}\left(1 - \frac{1}{2}\,\text{Erfc}\left[\frac{\text{mu}-\text{p}}{\sqrt{2}\ \text{sigma}}\right]\right)}$$

then set up the system of equations defining the given conditions.

In[5]:=
```
calibrationconditions = {choiceprob[25] == 0.2, elasticity[25] == -2}
```

Out[5]=
$$\left\{1 - \frac{1}{2}\,\text{Erfc}\left[\frac{-25 + \text{mu}}{\sqrt{2}\ \text{sigma}}\right] == 0.2,\ -\frac{25\,e^{-\frac{(-25+\text{mu})^2}{2\,\text{sigma}^2}}}{\sqrt{2\,\pi}\ \text{sigma}\left(1 - \frac{1}{2}\,\text{Erfc}\left[\frac{-25+\text{mu}}{\sqrt{2}\ \text{sigma}}\right]\right)} == -2\right\}$$

Now you may have to experiment some to find the parameters solving these conditions. You won't be able to solve for the parameters using Solve; this isn't an algebraic system of equations. So you have to use FindRoot. If you assigned the system of equations to a variable conditions for example, you would use the command FindRoot[conditions,{{mu,startmu},{sigma,startsigma}}] where startmu and start-sigma are an initial guess at a solution. You may have to try several combinations of values before FindRoot will give a solution. It may help to
Plot[1-CDF[NormalDistribution[startmu,startsigma],p],{p,10,40}],
say, to get a feel for the range of parameters.

In[6]:=
```
calibrationsol = FindRoot[calibrationconditions, {{mu, 20}, {sigma, 40}}]
```

Out[6]=
```
{mu → 10.2736, sigma → 17.4976}
```

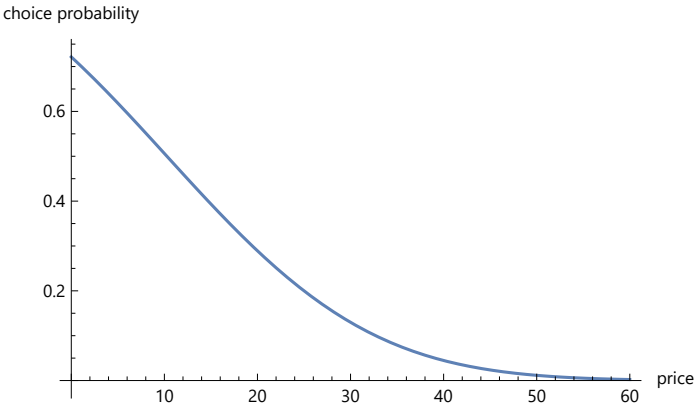
In[7]:=
```
{mu, sigma} = {mu, sigma} /. calibrationsol
```

Out[7]=
```
{10.2736, 17.4976}
```

If you get a satisfactory solution you can assign the solution values to the variables mu and sigma for the remaining problems, plot the final choice probability function, and check its value and elasticity at *p* = 25.

In[8]:=
```
Plot[choiceprob[p], {p, 0, 60}, AxesLabel → {"price", "choice probability"}]
```

Out[8]=



In[9]:=
```
choiceprobat25 = choiceprob[25]
```

Out[9]=
```
0.2
```

In[10]:=
```
elasticityat25 = elasticity[25]
```

Out[10]=
```
-2.
```

2. [Simulation] Suppose now that this choice probability function correctly predicts the probability that a random consumer buys at price p. Using the RandomVariate and Table functions, write a function that will generate a list of 100 random samples from the distribution of values determined in part 1. Using the following function to count the number of values in a list which are greater than a given p, write a simulation function sampledemand[p_] that will count how many of 100 random consumers will choose to buy at a given price p. For prices p from $10. to $40. in steps of $0.50 generate, using Table, a list of pairs of the form {p,sampledemand[p]}, giving the number of buyers at each price. Use ListPlot to visualize the data.

In[11]:=
```
numbervaluesgreater[s_, p_] := Length[Select[s, # > p &]];
```

Evaluate the above definition. Define a function to generate 100 samples, say sample:=... using := to make a definition of sample that will be reevaluated every time it is used.

In[12]:=
```
sample := Table[RandomVariate[dist], {100}]
```

Write the function sampledemand[p_]:=... , again with := so you get a new sample of values each time, counting the number of sample values greater than p. You might try your function to make sure that it gives you about 20 (=20% of 100) when evaluated at a price of $25, that it can give you different values at times, and that it generally gives you higher values if you decrease the price and vice versa.

In[13]:=
```
sampledemand[p_] := numbervaluesgreater[sample, p]
```

In[14]:=
```
sampledemand[25]
```

Out[14]=

```
11
```

Next build a table, using Table, of prices and sampledemand. Use demanddata=... with a single equal sign = now because you want to evaluate the right hand side, to generate the data just once. The way to specify a sequence of prices in a range with a specified step size between successive prices is to write a list {p,startp,endp,stepsize} as second argument to the Table command.

In[15]:=
```
demanddata = Table[{p, sampledemand[p]}, {p, 0, 60., 1}]
```

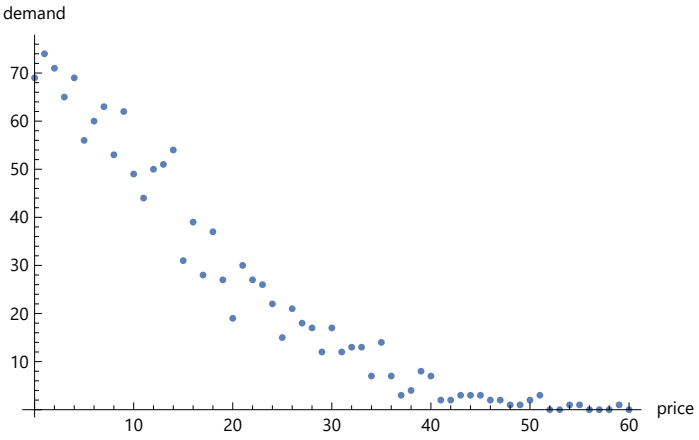
Out[15]=

```
{{0, 69}, {1, 74}, {2, 71}, {3, 65}, {4, 69}, {5, 56}, {6, 60}, {7, 63}, {8, 53}, {9, 62},
 {10, 49}, {11, 44}, {12, 50}, {13, 51}, {14, 54}, {15, 31}, {16, 39}, {17, 28},
 {18, 37}, {19, 27}, {20, 19}, {21, 30}, {22, 27}, {23, 26}, {24, 22}, {25, 15},
 {26, 21}, {27, 18}, {28, 17}, {29, 12}, {30, 17}, {31, 12}, {32, 13}, {33, 13},
 {34, 7}, {35, 14}, {36, 7}, {37, 3}, {38, 4}, {39, 8}, {40, 7}, {41, 2}, {42, 2},
 {43, 3}, {44, 3}, {45, 3}, {46, 2}, {47, 2}, {48, 1}, {49, 1}, {50, 2}, {51, 3},
 {52, 0}, {53, 0}, {54, 1}, {55, 1}, {56, 0}, {57, 0}, {58, 0}, {59, 1}, {60, 0}}
```

dataplot=ListPlot[demanddata] creates a plot of the data and assigns it to the variable dataplot (values of variables can also be graphs).

In[16]:=
```
dataplot = ListPlot[demanddata, AxesLabel → {"price", "demand"}]
```

Out[16]=



3. [Estimation] Imagine now that you were given the price and demand data you just generated in part 2. Fit a linear demand function approximating the data. Plot the linear fit and Show it with the data points graph you plotted. Compare with the graph of 100*choiceprob[p] which is the actual expected demand used in the simulation to generate the data.

The function Fit does a least squares approximation to data, using a list of functions {1,p} say, and the list of dependent variables, {p} in this case, as second and third arguments.

In[17]:=
```
demandFit = Fit[demanddata, {1, p}, {p}]
```

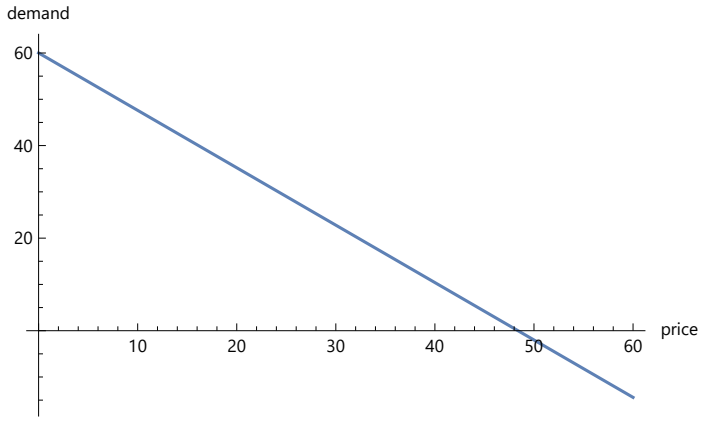
Out[17]=

$$60.0122 - 1.2403\ p$$

The plots below shows the above linear fit independently and with the scatter plot of demand data.

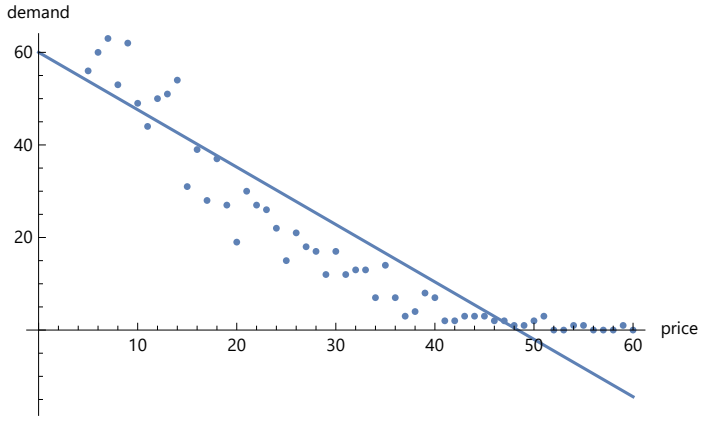
In[18]:=
```
demandplot = Plot[demandFit, {p, 0, 60}, AxesLabel → {"price", "demand"}]
```

Out[18]=



In[19]:=
```
Show[demandplot, dataplot]
```

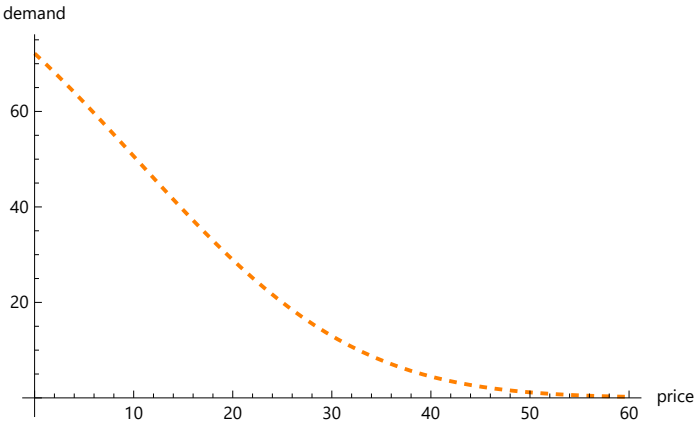
Out[19]=



Next we plot the expected demand curve using the choice probability function, and compare it with the above obtained linear fit.

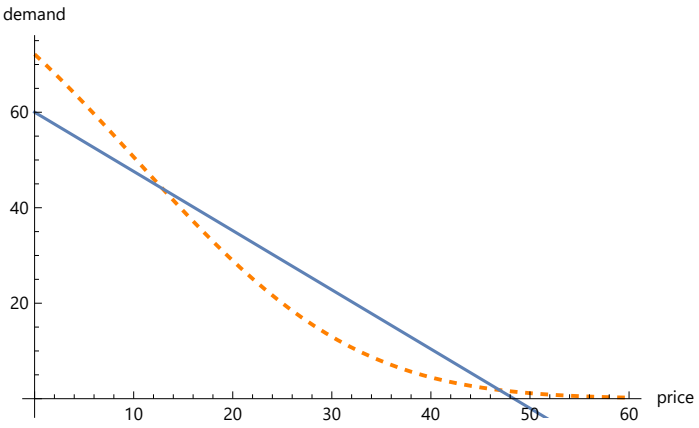
In[20]:=
```
choiceprobplot = Plot[100 * choiceprob[p], {p, 0., 60.},
  PlotStyle → {Orange, Dashed, Thick}, AxesLabel → {"price", "demand"}]
```

Out[20]=



In[21]:=
```
Show[choiceprobplot, demandplot]
```

Out[21]=



4. [Comparison] Compute the profit maximizing price for a firm with marginal cost of 15. using first the linear demand fit, and second with the expected demand defined by the choice probability function.

In[22]:=
```
mc = 15.;
```

We define two profit functions, one based on the linear fit, and the other based on the expected demand defined by the choice probability function.

In[23]:=
```
profit1[p_] := p * demandFit - mc * demandFit
```

In[24]:=
```
profit2[p_] := 100 * (p * choiceprob[p] - mc * choiceprob[p])
```

The profit is maximized at the critical points.

In[25]:=
```
profitmaxcondition1 = D[profit1[p], p] == 0
```

Out[25]=

$$78.6166 - 2.48059\, p == 0$$

In[26]:=
```
profitmaxcondition2 = D[profit2[p], p] == 0
```

Out[26]=

$$100\left(1+0.341997\,e^{-0.0016331\,(10.2736-p)^2}-0.0227998\,e^{-0.0016331\,(10.2736-p)^2}\,p-\frac{1}{2}\,\text{Erfc}\left[0.0404116\,(10.2736-p)\right]\right)==0$$

Solving for the critical points gives the profit maximizing price.

In[27]:=
```
profitmaxprice1 = Solve[profitmaxcondition1, p][[1]]
```

Out[27]=

$$\{p\to 31.6927\}$$

In[28]:=
```
profitmaxprice2 = FindRoot[profitmaxcondition2, {p, 30.}]
```

Out[28]=

$$\{p\to 26.8116\}$$

The maximum profits obtained as predicted by each profit function is now computed.

In[29]:=
```
maxprofit1 = profit1[p] /. profitmaxprice1
```

Out[29]=

345.603

In[30]:=
```
maxprofit2 = profit2[p] /. profitmaxprice2
```

Out[30]=

203.502