Potential Welfare Impacts From the Continued Spread of Wild Pigs

Jason Holderieath[1], Michael Crosby[2], Eric McConnell[2]
[1]Corresponding Author: School of Agricultural Sciences and Forestry, Louisiana Tech University, jjhold@latech.edu
[2]School of Agricultural Sciences and Forestry, Louisiana Tech University

Selected Paper prepared for presentation at the Southern Agricultural Economics Association (SAEA) Annual Meeting, Birmingham, Alabama, February 2-5, 2019

Potential Welfare Impacts From the Continued Spread of Wild Pigs

Jason Holderieath, Michael Crosby, Eric McConnell

Abstract

Wild Pigs are spreading across the United States and bringing damage with them. Damage estimates from a survey of producers reported 2014 crop losses of $190 million in 11 southeastern US states (Anderson et al. 2016), and associated short-run welfare losses were calculated at $142 million (Holderieath et al. 2018). With approximately 80% of corn, 78% of soybeans, 62% of wheat, 6% of rice, and 1% of peanuts produced in counties that do not have wild pigs, there is a substantial amount of production at risk in the event of continued spread. McClure et al. (2015) and Snow, Jarzyna, and VerCauteren (2017) have demonstrated that wild pigs can survive in much of the US; however, not all US counties are equally likely to be invaded. This work integrates the probability of wild pig invasion to predict the likely welfare effects of wild pigs continuing to spread using a random forest based machine learning model linked to an equilibrium displacement model. Results from this iteration show a likely increase in total surplus. However, this result should be interpreted with caution due to the relatively low number of observations and the need for further analysis, as well as the multitude of other causative factors.

Introduction

Invasive wild pigs, also known as feral swine and wild boars (*Sus scrofa*), were introduced to the Southeastern United States in the 16th century by Spanish explorers (Keiter, Mayer, and Beasley 2016). Wild pig presence can be costly as they are known to carry diseases dangerous to humans and livestock, depredate and compete for resources with native wildlife, and damage property. Total damages have been estimated at $800 million per year Pimentel, Zuniga, and Morrison (2005). Direct production losses to corn, soybeans, wheat, rice, and peanuts in 10 of the affected states were estimated at $190 million per year through a producer survey (Anderson et al. 2016).

However, there is more to be considered than simple production losses. Production losses will affect prices, which will affect consumers and producers who do not experience damage. Net short-run welfare losses due to wild pig damage to corn, soybeans, wheat, rice, and peanuts were calculated at $142 million (Holderieath et al. 2018). However, three of those crops are primarily grown outside of the counties with known wild pig presence.

If wild pigs continue to spread, that absence may be short-lived. In recent years, wild pigs are spreading at an increasing rate across the continental United States. Over the 30 years between 1982 and 2012, the northward rate of expansion was 8.9 kilometers per year, and the yearly average rate of northward expansion from 2009 to 2012 was 12.6 kilometers per year (Snow, Jarzyna, and VerCauteren 2017). Building on the methods of Snow, Jarzyna, and VerCauteren (2017) and Holderieath et al. (2018), we paired an ecological model of the probability of spreading wild pigs with an economic model of crop damage to estimate the potential for welfare losses if wild pigs continue their northward spread.

Methods

Our basic approach was to estimate a similar model to Snow, Jarzyna, and VerCauteren (2017) with many of the same regressors and with the same end of predicting the probability of invasion in a period. However, we took a different approach to estimation, discussed in the next subsection. In three timesteps over 24 years, counties were randomly presented for a new infestation of wild pigs. Newly present wild pigs then affected the probability of neighboring counties when they were presented. In this way, the probability of spread was a function of the presence of wild pigs in neighboring and nearby counties in addition to regressors such as weather and land use. Once counties were presented for invasion, a random level of damage from a triangle distribution was assigned to the county. In modern times, wild pigs are rarely removed from a county, so the simulation did not provide for that. Another feature of modern invasion is likely human release at a distance from known populations. This model allowed for spontaneous invasion, given the likelihood developed from the model. Once wild pigs were present, they remained present for the duration of the simulation. Each county's production was sent to a national market, and price changes were calculated with an equilibrium displacement model. Welfare measures are calculated as changes in producer and consumer surplus.

Probability of Invasion

Snow, Jarzyna, and VerCauteren (2017) used an openBUGS (Bayesian inference Using Gibbs Sampling) model (OpenBUGS 2018) implemented in R (R Core Team 2018) for their prediction of wild pig territory expansion. Spatio-temporal presence data is available (Lutman 2013; Snow, Jarzyna, and VerCauteren 2017). However, the intervals are not even, and the data does not specify absence, only presence. Any attempt to use this data as a panel data set for estimation of spread will encounter both spatial and temporal non-stationarity. The non-stationarity should be expected because spatial nearness to wild pigs is one of the best predictors of the presence of wild pigs. The time steps are uneven, and eradication is rarely observed leading to suspicion over the time dimension. These problems could be corrected with use of a consistent estimator (e.g., a within or first differences econometric model).

The problem at hand is essentially an ecological classification problem—are wild pigs known to be present? This type of problem has been addressed with a machine learning method known as Random Forest (RF) (Cutler et al. 2007; Walsh et al. 2017). One particularly attractive characteristic of this approach is the lack of reliance on distributional assumptions (Cutler et al. 2007; Walsh et al. 2017), meaning that the non-stationarity across spatial and temporal dimensions is not a problem. At its most simple, the software builds numerous weighted decision trees and uses those trees to predict an outcome. The intuition of decision trees is also attractive as we can see the importance of variables on the predictions and understand that distance to the nearest known population of wild pigs is an important predictor of wild pig presence, for example (Cook 2017).

Data from the online appendix of Snow, Jarzyna, and VerCauteren (2017) was reshaped to standard panel format with variables in columns and observations across time in rows. Variables for precipitation, stream distance, road miles, and land-use were used in the analysis.

However, variables relating to temperature were removed due to an apparent loss of prediction accuracy. Temperature is highly correlated with latitude, and the RF appeared to put too much weight on those variables, reducing accuracy in predicting presence in later periods as wild pigs move north.

A more spatially direct modeling technique such as geographically weighted regression or kriging would not need data on distance to nearest known wild pigs because it would be implicit in the model. Adopting the RF approach meant a need to know how far wild pigs are from a given county (that is not itself). The distance between each county within 500 miles (Roth 2014) was combined with wild pig presence data (Lutman 2013) to create a minimum distance to wild pigs from each county variable to pair with the reshaped Snow, Jarzyna, and VerCauteren (2017) data.

The final list of variables used in the RF model of wild pig spread was "AREA_KM", "ECO_DIVISN", "mammal richness", "P_Ag", "P_Dev", "P_For", "P_Oth", "P_Ran", "P_Wat", "P_Wet", "A_MN_Ag", "A_MN_Dev", "A_MN_For", "A_MN_Oth", "A_MN_Ran", "A_MN_Wat", "A_MN_Wet", "A_CV_Ag", "A_CV_Dev", "A_CV_For", "A_CV_Oth", "A_CV_Ran", "A_CV_Wat", "A_CV_Wet", "PD", "AREA_AM", "AREA_CV", "CWED", "CONTAG", "IJI", "DIVISION", "SIDI", "AI", "Annual_Precip", "Summer_Precip", "Winter_Precip", "maxPrecip", "minPrecip", "STREAM_KM", "ROAD_KM", "HumanPop", “precipsum”, “precipwin”, “popdes.slope”, “mi_to_county,” all except for “mi_to_county” were from Snow, Jarzyna, and VerCauteren (2017).

A software product, H2O (Cook 2017), was used to implement the RF in R (R Core Team 2018). Training was carried out on the periods 1982-1988, 1988-2004, and 2009-2012. A Cross-Validation method with fifty folds was used to validate the model. The model was able to predict presence or unknown with approximately 9% error. The model was tested with the period 2004-2009 and yielded slightly higher errors of approximately 10%. Output including the confusion matrix is available in Appendix 1.

Spatio-temporal Simulation of Spread

With an estimated probability of invasion function, counties were evaluated for the probability of invasion in random order, without replacement. The RF model developed in the previous section was used to evaluate iteratively updated data on wild pig presence. Random presentation, rather than south to north was an attempt to simulate human introduction. The county’s probability of invasion was evaluated, then a random draw from *u(0,1)* was compared to the probability of invasion. If the probability is greater than the draw, the county was invaded, and its status was updated. Time-steps were set at an attempt to average the wildly varying periods between the observation points of 1982, 1988, 2004, and 2012. Predicted observations were set for 2020, 2028, and 2036 for a long-term outlook on the problem. Intervals to simulate the short-run would have yielded stronger results. However, the observational data of wild pig spread across short-run periods does not exist.

Price Changes

Price changes due to the invasions were found with an equilibrium displacement model (equation 1). Equilibrium displacement models (EDM) begin with the premise that the market in question is in equilibrium, the market is shocked, and then moves to another equilibrium (Nogueira et al. 2015; Holderieath et al. 2018; Brester, Marsh, and Atwood 2004). For each of the five crops,

$$EP_k * \eta_{kk}^{D} + \sum_{j=1}^{J}(EP_k * \eta_{kj}^{D}) + s_k^{Exports} * EP_k * \eta_{kk}^{Exports} = s_k^{Imports} * EP_k * \eta_k k^{Imports} + \sum_{fips=1}^{FIPS}(s_k^{fips} * (EP_k * \eta_{kk}^{fips} + \sum_{j=1}^{J}(EP_k * \eta_{kj}^{fips}) + EB_k^{fips}))) \quad (1)$$

was solved simultaneously with the change operator (E) modifying the price of the commodity, EP_k, is used to denote the relative change of the price of commodity (k), elasticities (η) of demand (D), $exports$, $imports$, and production locations ($fips$) for own price (kk) and cross price elasticities (kj), production and consumption weights (s) with the same notation, and exogenous relative production shocks (EB).

Elasticities were the same as used by Holderieath et al. (2018), except unit elasticities were used for import and export markets and additional supply elasticities originated from FAPRI-MU (2004). Verification of the model was conducted with unit own price elasticities and no cross-price elasticities. Price changes with this set of verification elasticities were positive as expected from a restriction of supply.

The only market level analyzed was at the farm production level, because that is where the damage occurs. Exogenous production shocks were randomly drawn from a triangle distribution with the lower limit and upper limit from the minimum and maximum values for each crop reported in Anderson et al. (2016) for each invaded county each period. The midpoint parameter for the triangle distribution was simply the center of the difference between the lower and upper values. There is substantial variation between states reported by Anderson et al. (2016), and it is not immediately apparent what states would be most alike with respect to wild pig damage drivers. Nor is it likely that counties are equally impacted within states. There were no consumption shocks in the model, so they have been omitted for clarity.

As a matter of programming pragmatism, it was easier to solve this set of equations in the form $Ax = b$. Five price changes, x, were solved by simplifying to a $5x5$ matrix of weighted elasticities, A, and a $5x1$ matrix of weighted exogenous shocks. This was accomplished with some rearranging.

The demand side for corn is expanded below as an example. First, notice that the domestic consumption does not have a weight. This is due to the derivation of the EDM not including a weight for domestic consumption. Second, notice that the export demand equation does not include terms for cross-price elasticities. The specification of the EDM did not include cross-price elasticities of export demand. Third, take note that there are no exogenous shocks included on the demand side. This was assumed in specification.

$$\begin{aligned}
&\mathrm{EP}_{corn} * \eta^{\mathrm{D}}_{\mathrm{corn,corn}} + \\
&\mathrm{EP}_{\mathrm{soy}} * \eta^{\mathrm{D}}_{\mathrm{corn,soy}} + \\
&\mathrm{EP}_{\mathrm{wheat}} * \eta^{\mathrm{D}}_{\mathrm{corn,wheat}} + \\
&\mathrm{EP}_{\mathrm{rice}} * \eta^{\mathrm{D}}_{\mathrm{corn,rice}} + \\
&\mathrm{EP}_{\mathrm{peanuts}} * \eta^{\mathrm{D}}_{\mathrm{corn,peanuts}} + \\
&\mathrm{s}^{\mathrm{Exports}}_{\mathrm{corn}} * \mathrm{EP}_{\mathrm{corn}} * \eta^{\mathrm{Exports}}_{\mathrm{kk}}
\end{aligned} \quad (2)$$

Then the supply side. Again, import supply is specified without a shock or cross-price elasticities, however, in practice, they are assigned zero values. For the sake of compactness, each county's supply is indexed by the $fips$ identifier. The county weight is distributed.

$$s_k^{Imports} * \ EP_{corn} * \eta_{corn,corn}^{Imports} + \sum_{fips}^{FIPS} (s_k^{fips} \ * EP_{corn} * \ eta_{kk}^{fips} + s_{corn}^{fips} \ * (EP_{soy} * \ \eta_{corn,soy}^{fips}) + s_{corn}^{fips} \ * (EP_{wheat} * \ \eta_{corn,wheat}^{fips}) + s_{corn}^{fips} * (EP_{rice} \ * \ \eta_{corn,rice}^{fips}) + s_{corn}^{fips} \ * (EP_{peanuts} * \ \eta_{corn,peanuts}^{fips}) + s_{corn}^{fips} \ * EB_{corn}^{fips})) \tag{3}$$

Once assembled as five very long equations without parentheses, the terms on the supply side multiplied by the EP_k terms were then moved to the demand side by multiplying them by -1. The numerical coefficients being multiplied by the EP_k terms were collected together by addition. Price changes were found by solving $Ax = b$ for x.

Presented in line with the development of this model is a test version with unit own price elasticities, cross price elasticities of zero, the same weights, and a random draw of exogenous shocks (Table 1). As one looks at each market individually, one should expect that additional destruction of crops by wild pigs will result in an inward shift of the supply curve. The inward shift should result in higher prices and lower quantity. The test case of unit elasticities with no cross-price elasticities is an opportunity to evaluate the theoretical performance of the equilibrium displacement model. Once solved, the model performed as expected yielding positive price changes (Table 2). As expected, quantities decreased (Table 3).

Welfare Effects

Welfare changes were measured geometrically and summed. The sensitivity of these types of shifts to functional form is minimal, and the error imposed by using linear approximations should be acceptable so long as the changes are relatively small (Brester, Marsh, and Atwood 2004; Alston, Norton, and Pardey 1995). Welfare changes from the initial time-step to the last are summed to find total damage over the simulation period. Only long-run elasticities are used, and the adjustment is assumed to be complete within each time-step. Welfare changes are calculated as changes in producer and consumer surplus. Consumer surplus change is calculated

$$\int_0^{Q_1} f\,(Q)dQ - P_1 \times Q_1^D - \int_0^{Q_0} f\,(Q)dQ - P_0 \times Q_0^D. \tag{4}$$

where Q_1 and P_1 represent the quantity and price clearing the market after the exogenous shock and Q_0 and P_0 represent the quantity and price prior to the shock.

Evaluation of the EDM yielded modest consumer surplus changes. The shift between 2012 and 2020 resulted in a loss of \$63,666.42. Shifts between 2020 to 2028 and 2028 to 2036 resulted in consumer surplus losses of \$135,599 and \$68,045.70, respectively.

A priori it is unknown if the decrease in quantity will overcome the increase in price to make producers worse off. Corn producers in the evaluation scenario were better off in all years: \$43,831,154, \$63,481,915, \$47,210,212 in 2020, 2028, and 2036 respectively. Soybean and wheat producers also gained ground (\$11,117,650, \$17,582,688, \$8,333,845 and \$6,610,204, \$9,473,328, \$5,268,545, respectively). As one might expect, rice producers were not impacted much because most rice-producing counties were already impacted by wild pigs (\$358,479, \$75,647.27, \$18,645.66).

Peanut producers were also not impacted very much for the same reasons as rice producers ($174,743, $67,254.90, $0).

Net surplus gains or losses are one indicator of the appropriateness of engaging in costly removal processes. For the 2020, 2028, and 2036 periods the total surplus change was $62,028,563, $90,545,234 and $60,763,202, respectively. This would seem to indicate that the spread of wild pigs will increase welfare. However, the story is much more complex than that. This was a single set of damage values for a single spread scenario. The picture becomes clearer with the full set of elasticities and several simulations with different spread and damage scenarios presented in the next section. However, we can tell based on these results that the equations are working properly, allowing us to move on to the next step.

Results and Discussion:

A single observation as presented in the Methods section is not an adequate exploration of the topic. The uncertainty surrounding what counties will be invaded and what damage will be likely in those counties suggests that a simulation is in order. To that end, the prediction model and price change models were run 100 times. For this draft of the paper, 100 simulations were chosen due to the approximately 5.83 hours required for a preliminary run on a Linux system with a GUI (5.25 hours without the GUI) required to run the model 100 times. Future drafts will increase the number of runs.

In each panel, Figure 1 shows the percent of simulations in which wild pigs were present in each county by the captioned year. Recall that once established, wild pig populations could not be extirpated and that the probabilistic nature of innovation allowed for the non-continuous spread of wild pigs. The resulting map shows the probability of presence in each county in each period. The projected spread is aggressive, however, given the spread over the past 30 years not unreasonable. As one would expect, the spread will mostly be continuous. However, some probable pockets appear to have developed.

Price changes with the full set of elasticities over the 100 simulations were spread across a relatively narrow range within each crop, excepting peanuts (Figure 2). Corn prices moved downward in all three periods in all simulations. Soybean price changes were very small, mostly centered on no change. Wheat prices were mostly negative, excepting the top 10% to 15% of price changes. Rice prices increased between 0% and 0.25%. Peanut prices were substantially more variant increasing across the 0% to 1.5% range.

With the exception of corn, which had the most negative price movements, and wheat for a very small proportion of simulations consumer surplus was generally negative across periods (Figure 3, Figure 4, and Figure 5). Sum of change in consumer surplus across commodities was only positive in the last period. As domestic consumers of these products are facing a single market, there were no subgroups below this distribution to which the model can speak.

Producer surplus in Figure 6, Figure 7, and Figure 8 are summed across counties, so this change in producer surplus represents the change of fortunes of producers as a whole. It is important to remember that between time-steps the entire market adjustment is completed. Only counties that have new wild pig damage would be expected to experience an exogenous shock. Change in producer surplus in counties with wild pig damage in previous periods will primarily be driven by price changes and secondarily changes in damage levels. Further analysis is needed to separate the effects of the spread between affected and unaffected counties. The positive change in corn producer surplus is enough to overwhelm the modest and negative changes in other commodities to net a large positive change in producer surplus.

Total surplus gains are modest and centered approximately on zero for the first two periods followed by a very large expected gain in total surplus in 2036 across simulations (Figure 9). The difference between 2028 and 2036 is likely explained by the number and productivity of the counties invaded in that period.

It would be reasonable to look at these results and suggest that allowing the spread may be beneficial. That is likely not the case. Practically speaking, only five crops are covered by the analysis. There are numerous other dimensions of costly damage inflicted by wild pigs. Future iterations of this work will investigate how to restrict supply beyond the use of elasticities. Cross price elasticities allow for substitution of production, which may be overstated. Further, we do not know if planting behavior changes with introduction of wild pigs. These are all avenues of elaboration for this work.

As proof of concept, this model succeeds in using an accurate machine learning algorithm to predict the spread of wild pigs and then using that spread to determine welfare effects. Further research needs to examine factors leading to the pockets of probable wild pig spread that were not at the margin of an existing population. Further, examination factors leading to past non-continuous introduction and incorporating those factors explicitly into the spread model, possibly including factors not included in the

Snow, Jarzyna, and VerCauteren (2017) analysis. More observations are needed. Future work will streamline the programming and increase observations.

The analysis also begs the question, can the EDM be eliminated from the process? Only wild pig spread and damage was treated as uncertain in these scenarios. However, there is also uncertainty surrounding elasticities. It seems reasonable that a sufficiently sophisticated machine learning model could be used to predict damage with little loss in the ability to understand factors predicting that damage.

References

Alston, Julian M, George W. Norton, and Philip G. Pardey. 1995. *Science under Scarcity Principles and Practice for Agricultural Research Evaluation and Priority Setting*. Edited by George W Norton and Philip G Pardey. Food Systems and Agrarian Change. Ithaca: Cornell University Press.

Anderson, Aaron, Chris Slootmaker, Erin Harper, Jason Holderieath, and Stephanie A Shwiff. 2016. "Economic Estimates of Feral Swine Damage and Control in 11 US States." *Crop Protection* 89 (November):89–94. https://doi.org/http://dx.doi.org/10.1016/j.cropro.2016.06.023.

Brester, Gary W, John M Marsh, and Joseph A Atwood. 2004. "Distributional Impacts of Country-of-origin Labeling in the U.S. Meat Industry." *Journal of Agricultural and Resource Economics* 29 (2):206–27.

Cook, Darren. 2017. *Practical Machine Learning with H2O*. Edited by Nicole Tache, Colleen Lobner, and Kim Cofer. First. Sebastopol, CA: O'Reilly Media, Inc.

Cutler, D. Richard, Thomas C Edwards, Karen H Beard, Adele Cutler, Kyle T Hess, Jacob Gibson, and Joshua J Lawler. 2007. "RANDOM FORESTS FOR CLASSIFICATION IN ECOLOGY." *Ecology* 88 (11):2783–92. https://doi.org/10.1890/07-0539.1.

FAPRI-MU. 2004. "Docu mentation of the FAPRI Modeling System."

Holderieath, J.J., D.L. Pendell, J.C. Hadrich, A. Anderson, C. Slootmaker, E. Harper, and S.A. Shwiff. 2018. "Valuing the Absence of Feral Swine in the United States: A

Partial Equilibrium Approach." *Crop Protection* 112 (October). Elsevier:63–66. https://www.sciencedirect.com/science/article/pii/S026121941830125X.

Keiter, David A, John J Mayer, and James C Beasley. 2016. "What Is in a 'Common' Name? A Call for Consistent Terminology for Nonnative Sus Scrofa." *Wildlife Society Bulletin* 40 (2):384–87. https://doi.org/10.1002/wsb.649.

Lutman, Mark W. 2013. "Dataset: Feral Swine Presence."

McClure, M L, C L Burdett, M L Farnsworth, M W Lutman, D M Theobald, P D Riggs, D A Grear, and R S Miller. 2015. "Modeling and Mapping the Probability of Occurrence of Invasive Wild Pigs across the Contiguous United States." *PLoS One* 10. https://doi.org/10.1371/journal.pone.0133771.

Nogueira, Lia, Joel Michalski, Thomas L. Marsh, and Vicki McCracken. 2015. "Welfare Implications of Washington Wheat Breeding Programs." *Journal of Agricultural and Applied Economics* 47 (02):147–74. https://doi.org/10.1017/aae.2014.7.

OpenBUGS. 2018. "Home Page." Webpage. 2018. http://www.openbugs.net/w/FrontPage.

Pimentel, David, Rodolfo Zuniga, and Doug Morrison. 2005. "Update on the Environmental and Economic Costs Associated with Alien{\textendash}invasive Species in the United States." *Ecological Economics* 52 (3):273–88. https://doi.org/http://dx.doi.org/10.1016/j.ecolecon.2004.10.002.

R Core Team. 2018. *R: A Language and Environment for Statistical Computing*. *Vienna Austria R Foundation for Statistical Computing*. Vol. 1. Vienna, Austria: R

Foundation for Statistical Computing. http://www.r-project.org.

Roth, Jean. 2014. “Distances between County That Are within 500 Miles.” Cambridge, MA: National Bureau of Economic Research. http://www.nber.org/data/county-distance-database.html.

Snow, Nathan P., Marta A. Jarzyna, and Kurt C. VerCauteren. 2017. “Interpreting and Predicting the Spread of Invasive Wild Pigs.” Edited by Céline Bellard. *Journal of Applied Ecology* 54 (6):2022–32. https://doi.org/10.1111/1365-2664.12866.

Walsh, Eric S., Betty J. Kreakie, Mark G. Cantwell, and Diane Nacci. 2017. “A Random Forest Approach to Predict the Spatial Distribution of Sediment Pollution in an Estuarine System.” *PLoS ONE* 12 (7):1–18. https://doi.org/10.1371/journal.pone.0179473.

Tables and Figures

Table 1. Exogenous Shocks Used in Verification of Equilibrium Displacement Model.

	2020	2028	2036
Corn	-0.00217	-0.00314	-0.00233
Soybeans	-0.00164	-0.00259	-0.00123
Wheat	-0.00206	-0.00294	-0.00163
Rice	-0.00069	-0.00014	-0.00004
Peanuts	-0.00036	-0.00014	0.00000

Table 2. Price Changes Found in Verification of the Equilibrium Displacement Model.

	2020	2028	2036
Corn	0.123%	0.179%	0.133%
Soybeans	0.071%	0.112%	0.053%
Wheat	0.078%	0.112%	0.062%
Rice	0.024%	0.005%	0.001%
Peanuts	0.018%	0.007%	0.000%

Table 3. Quantities Found in Verification of the Equilibrium Displacement Model.

	2020	2028	2036
Corn	10,317,951,844	10,317,933,682	10,317,920,200
Soybeans	2,179,910,263	2,179,905,360	2,179,903,902
Wheat	2,179,908,919	2,179,904,205	2,179,901,587
Rice	198,274,068	198,273,855	198,273,802
Peanuts	6,461,502,234	6,461,498,470	6,461,498,470

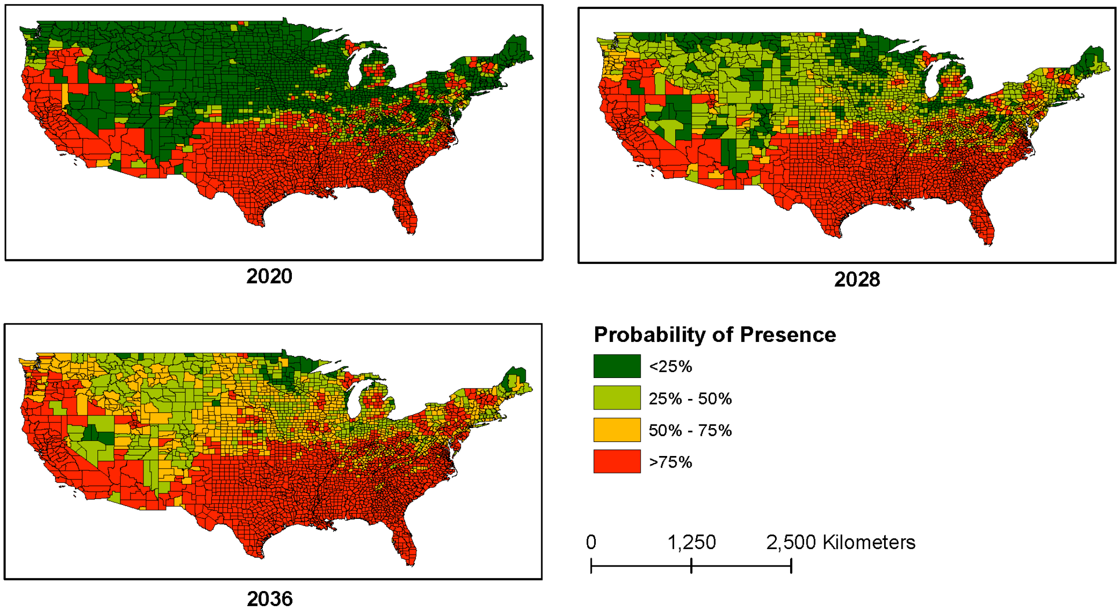

Figure 1. Probability of Presence

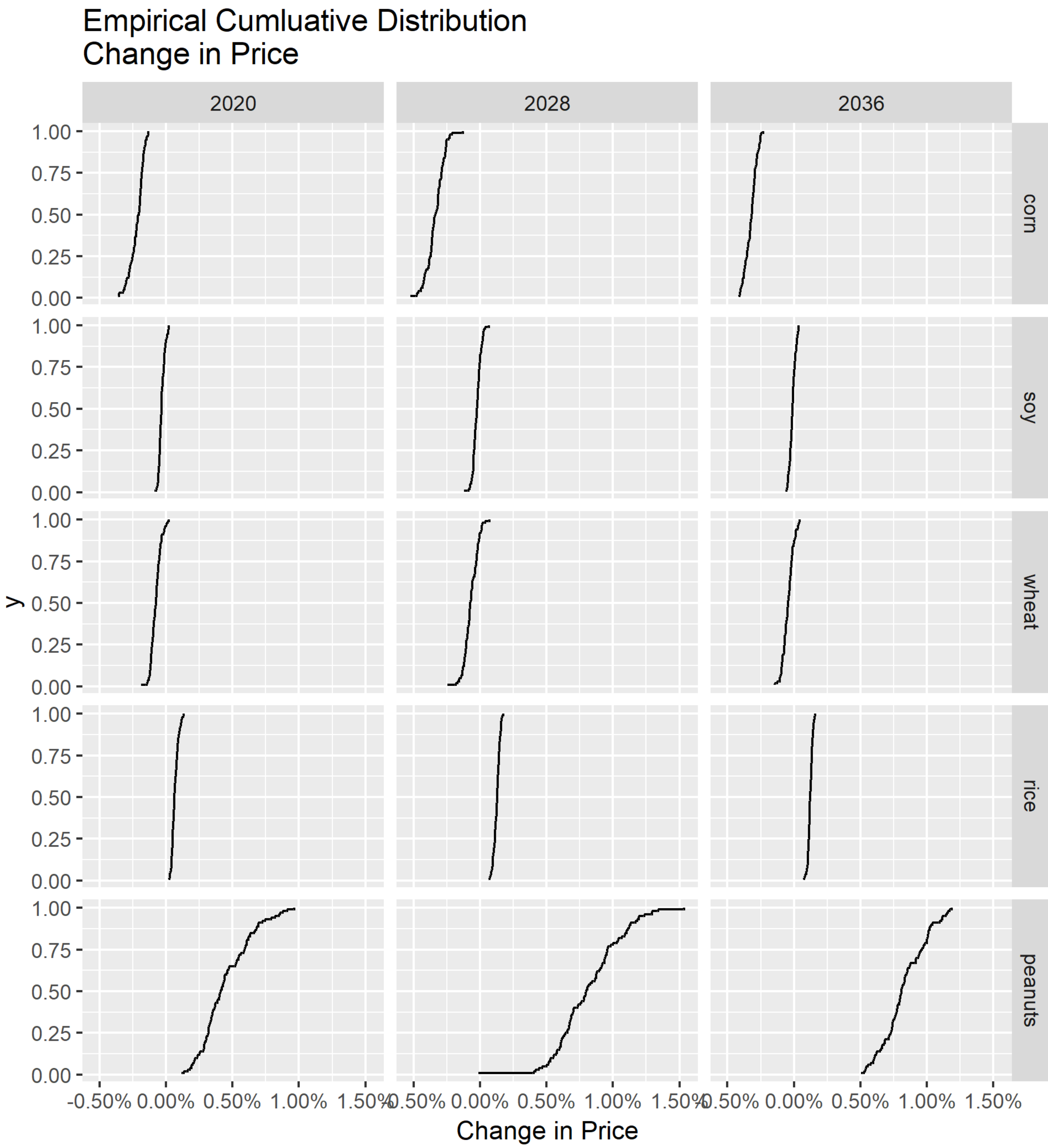

Figure 2. Empirical Cumulative Distribution of Changes in Price.

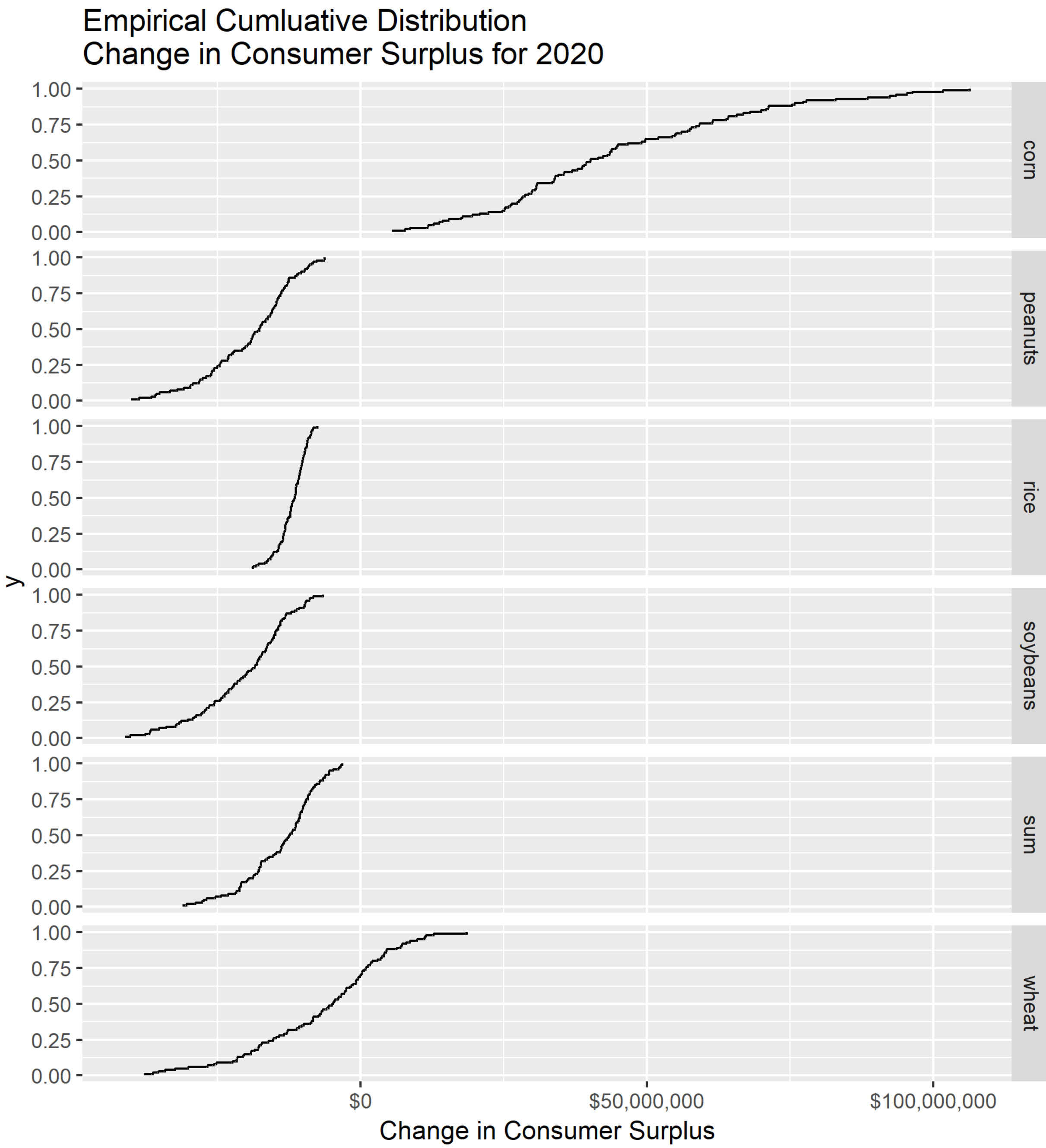

Figure 3. Empirical Cumulative Distribution of Changes in Consumer Surplus in Year 2020.

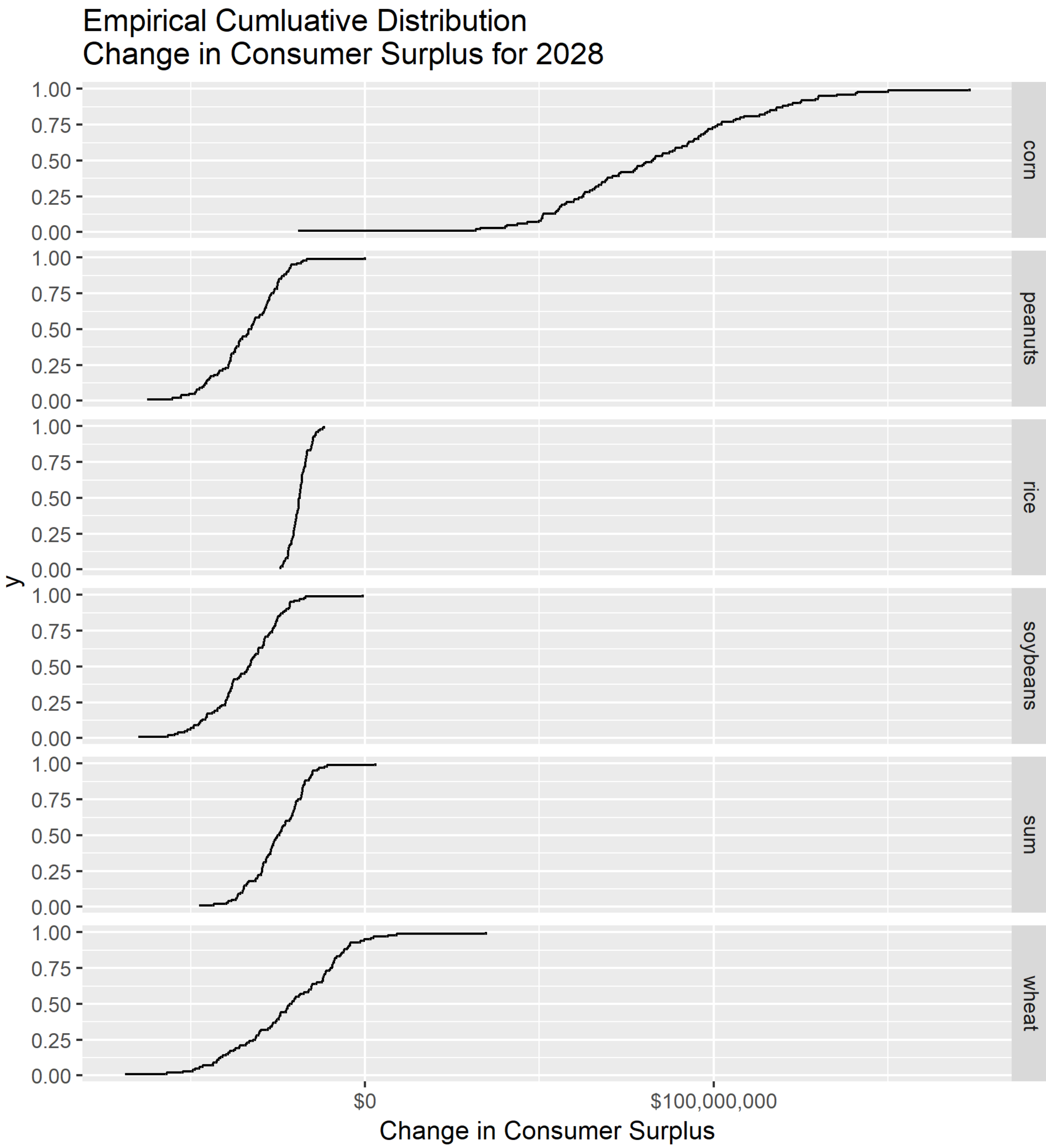

Figure 4. Empirical Cumulative Distribution of Changes in Consumer Surplus in Year 2028.

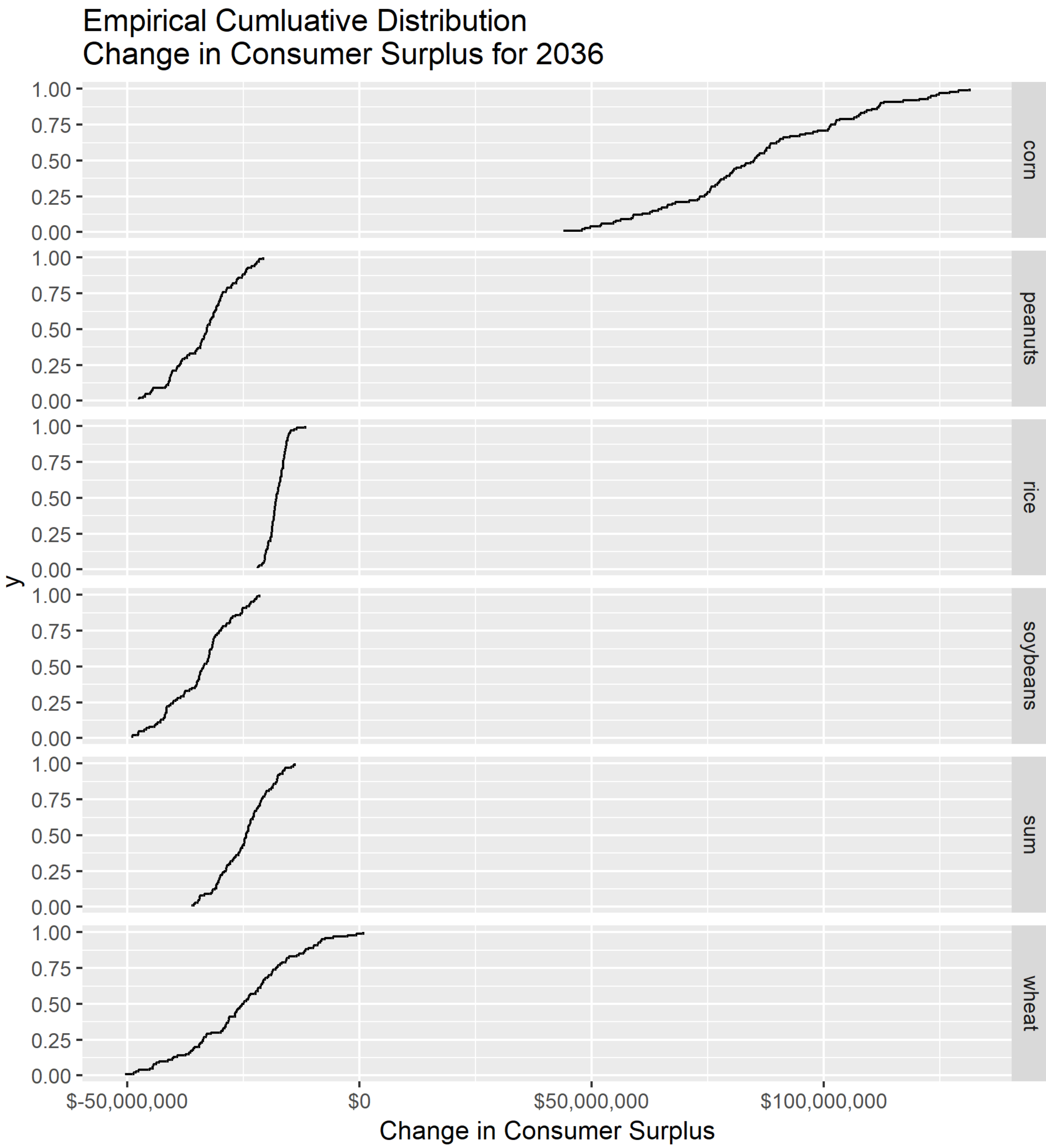

Figure 5. Empirical Cumulative Distribution of Changes in Consumer Surplus in Year 2036.

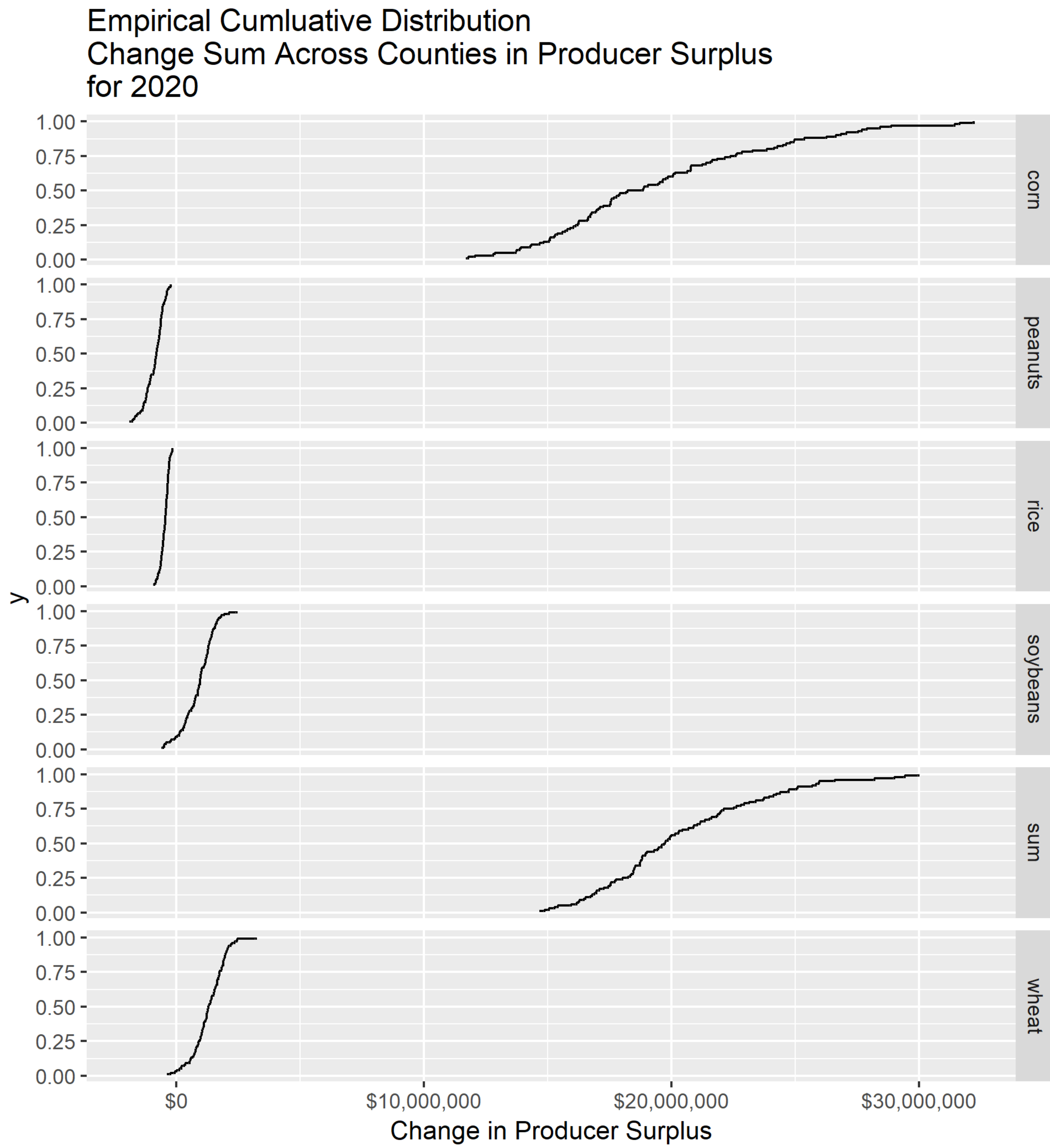

Figure 6. Empirical Cumulative Distribution of Changes in Producer Surplus in Year 2020.

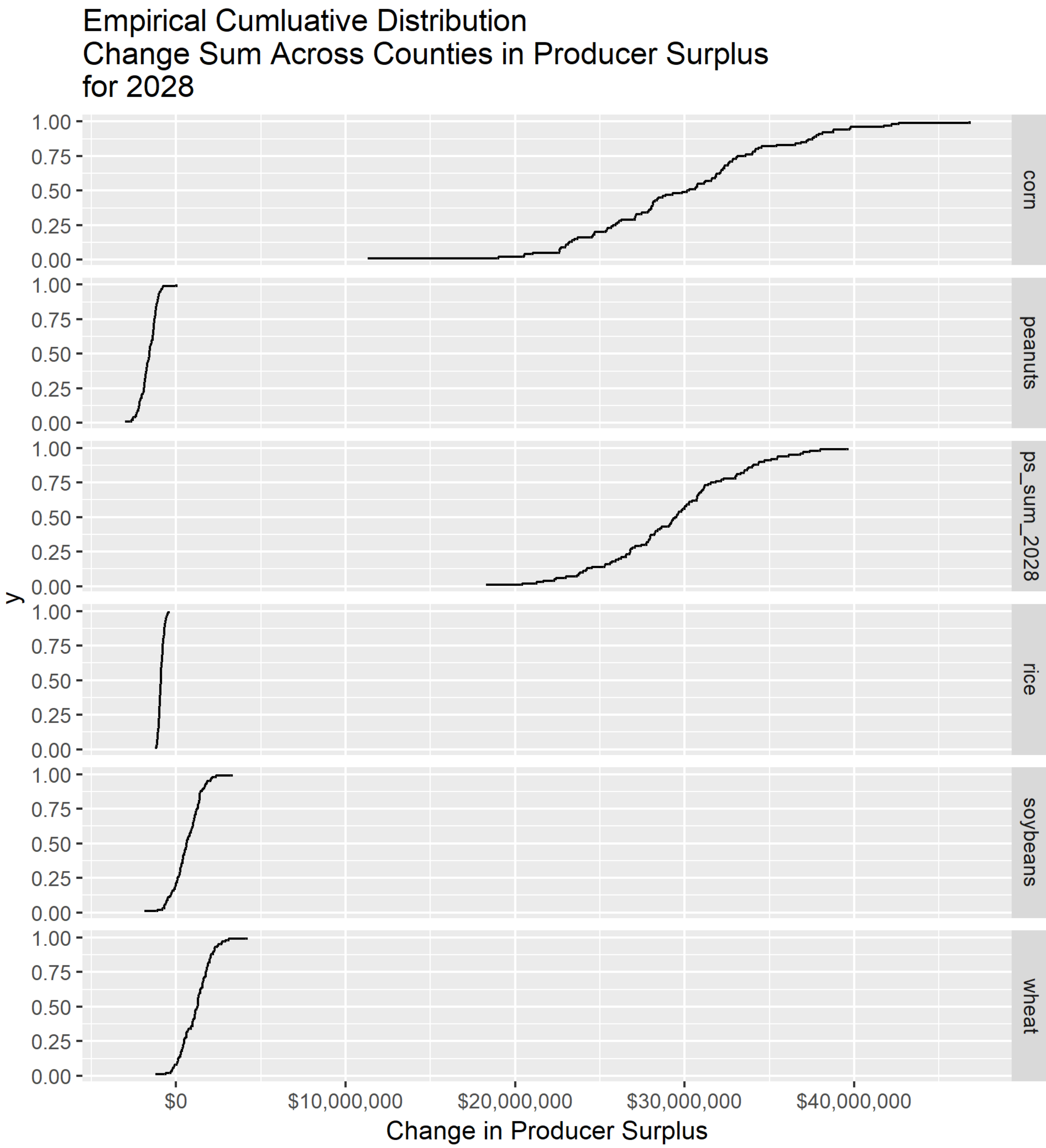

Figure 7. Empirical Cumulative Distribution of Changes in Producer Surplus in Year 2028.

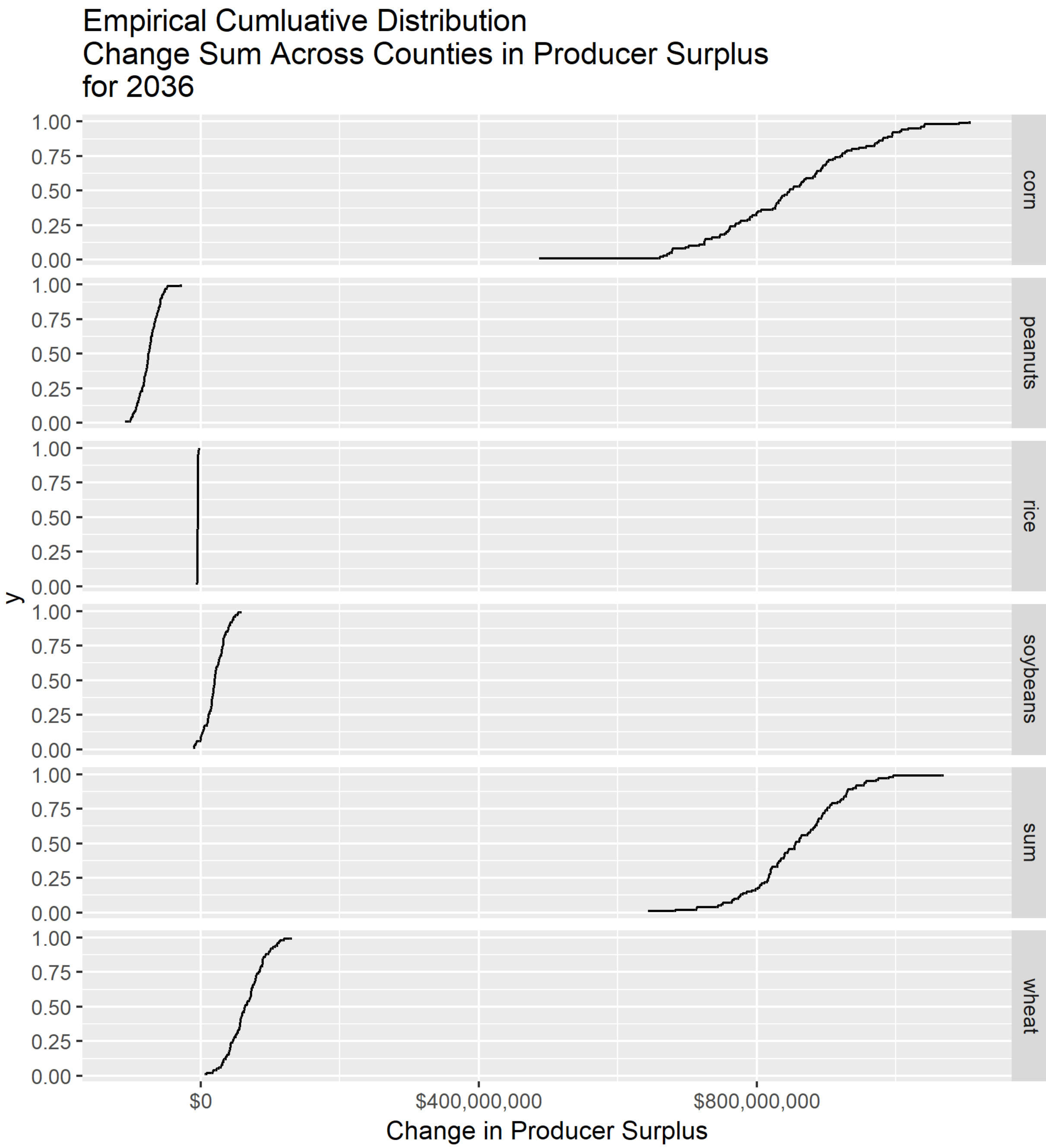

Figure 8. Empirical Cumulative Distribution of Changes in Producer Surplus in Year 2036.

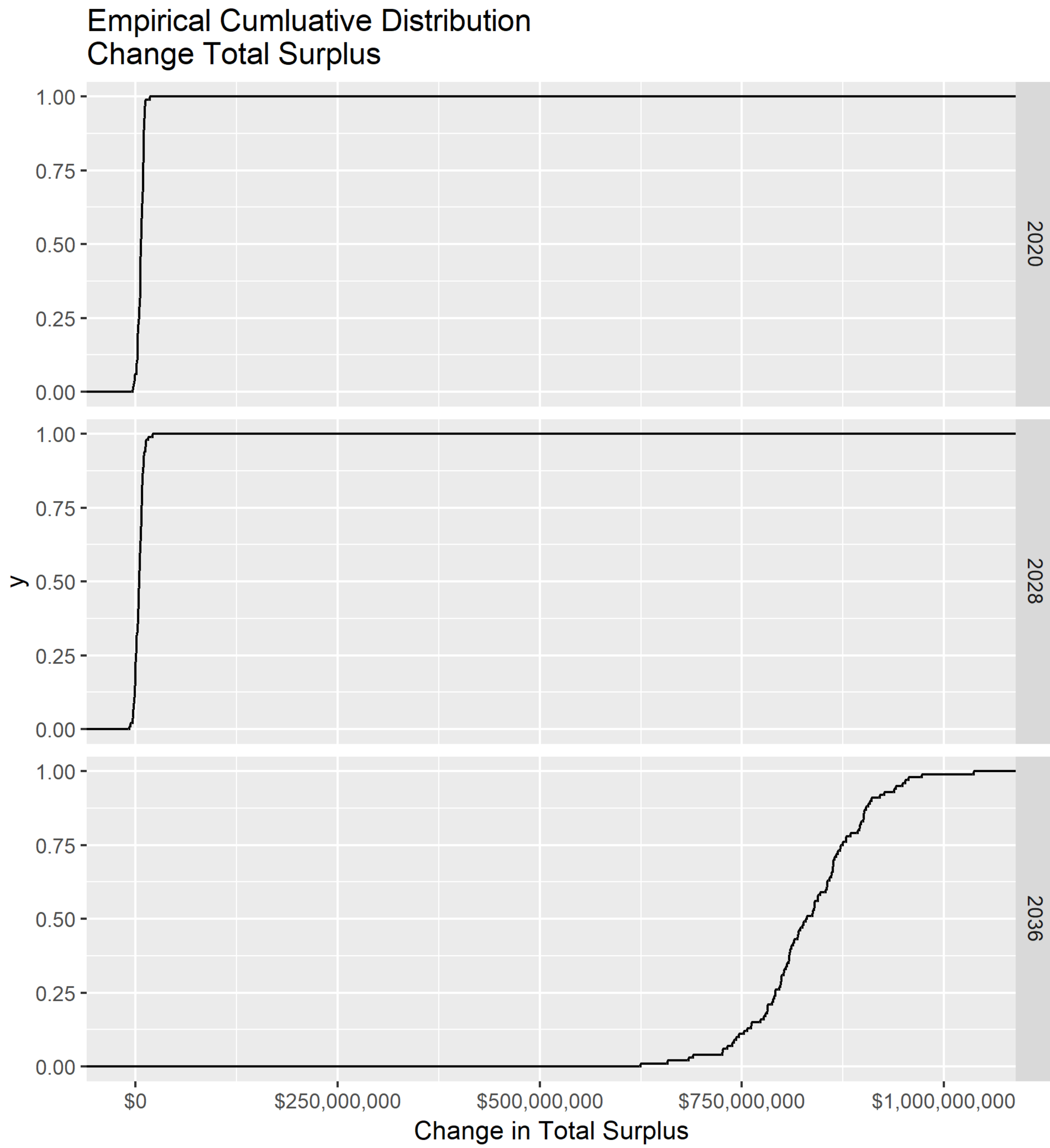

Figure 9. Empirical Cumulative Distribution of Changes in Total Surplus.

Appendix 1. Machine Learning Output

```
##
## H2O is not running yet, starting it now...
##
## Note:  In case of errors look at the following log files:
##
C:\Users\User\AppData\Local\Temp\Rtmp6l9liC/h2o_JHolderieath_started_from_r.o
ut
##
C:\Users\User\AppData\Local\Temp\Rtmp6l9liC/h2o_JHolderieath_started_from_r.e
rr
##
##
## Starting H2O JVM and connecting: . Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         2 seconds 288 milliseconds
##     H2O cluster timezone:       America/Chicago
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.20.0.8
##     H2O cluster version age:    3 months and 19 days !!!
##     H2O cluster name:           H2O_started_from_R_JHolderieath_lpq068
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   1.70 GB
##     H2O cluster total cores:    8
##     H2O cluster allowed cores:  8
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     H2O API Extensions:         Algos, AutoML, Core V3, Core V4
##     R Version:                  R version 3.5.0 (2018-04-23)

## Model Details:
## ==============
##
## H2OBinomialModel: drf
## Model ID:  RF_defaults
## Model Summary:
##   number_of_trees number_of_internal_trees model_size_in_bytes min_depth
## 1              50                       50              444856        19
##   max_depth mean_depth min_leaves max_leaves mean_leaves
## 1        20   19.98000        629        800   702.18000
##
##
## H2OBinomialMetrics: drf
## ** Reported on training data. **
## ** Metrics reported on Out-Of-Bag training samples **
```

```
##
## MSE:  0.06402607
## RMSE:  0.2530337
## LogLoss:  0.2691617
## Mean Per-Class Error:  0.09831253
## AUC:  0.9665403
## Gini:  0.9330807
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal
threshold:
##         present unknown    Error        Rate
## present    2476     412 0.142659  =412/2888
## unknown     347    6083 0.053966  =347/6430
## Totals     2823    6495 0.081455  =759/9318
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.508090 0.941277 224
## 2                       max f2  0.225485 0.957967 313
## 3                 max f0point5  0.636420 0.946718 182
## 4                 max accuracy  0.517544 0.918545 221
## 5                max precision  0.999994 0.996314   0
## 6                   max recall  0.000000 1.000000 399
## 7              max specificity  0.999994 0.996537   0
## 8             max absolute_mcc  0.517544 0.808586 221
## 9   max min_per_class_accuracy  0.647111 0.906856 178
## 10 max mean_per_class_accuracy  0.631689 0.909370 183
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or
`h2o.gainsLift(<model>, valid=<T/F>, xval=<T/F>)`
##
## H2OBinomialMetrics: drf
## ** Reported on cross-validation data. **
## ** 50-fold cross-validation on training data (Metrics computed for
combined holdout predictions) **
##
## MSE:  0.06150035
## RMSE:  0.2479926
## LogLoss:  0.2191091
## Mean Per-Class Error:  0.1043135
## AUC:  0.969902
## Gini:  0.939804
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal
threshold:
##         present unknown    Error        Rate
## present    2409     479 0.165859  =479/2888
## unknown     275    6155 0.042768  =275/6430
## Totals     2684    6634 0.080919  =754/9318
##
```

```
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.461860 0.942284 243
## 2                       max f2  0.260006 0.961970 305
## 3                 max f0point5  0.719964 0.950091 154
## 4                 max accuracy  0.515765 0.919725 224
## 5                max precision  0.999693 0.998758   1
## 6                   max recall  0.020013 1.000000 393
## 7              max specificity  0.999997 0.999307   0
## 8             max absolute_mcc  0.550210 0.811318 212
## 9   max min_per_class_accuracy  0.640000 0.909972 184
## 10 max mean_per_class_accuracy  0.633746 0.912356 186
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or
`h2o.gainsLift(<model>, valid=<T/F>, xval=<T/F>)`
## Cross-Validation Metrics Summary:
##                                 mean           sd  cv_1_valid  cv_2_valid
## accuracy                   0.9298214  0.012748753   0.9348837   0.9004525
## auc                       0.96970826  0.008113522  0.98580605  0.94908464
## err                       0.07017863  0.012748753  0.06511628  0.09954751
## err_count                       13.1    2.5661254        14.0        22.0
## f0point5                   0.9424099  0.014322855   0.9271978  0.89880955
## f1                        0.94980645  0.008932908   0.9507042  0.93209875
## f2                         0.9576857  0.010914464   0.9754335  0.96794873
## lift_top_group             1.4510386  0.052003957   1.5808823   1.4539474
## logloss                     0.219125  0.030309541  0.18164487  0.27059236
## max_per_class_error       0.14761259   0.04816932  0.16455697   0.3043478
## mcc                        0.8347905  0.030048912  0.86197037  0.76868683
## mean_per_class_accuracy   0.90855354  0.021074388  0.91404504   0.8445366
## mean_per_class_error      0.09144645  0.021074388  0.08595495  0.15546338
## mse                      0.061452143 0.0072631626 0.051195297 0.084659204
## precision                 0.93773067  0.019253572   0.9121622    0.877907
## r2                         0.7102924   0.03576039   0.7797373  0.60575515
## recall                    0.96322507  0.015760724  0.99264705   0.9934211
## rmse                      0.24702471  0.014678795  0.22626378  0.29096255
## specificity                 0.853882  0.049529858    0.835443   0.6956522
##                           cv_3_valid  cv_4_valid  cv_5_valid  cv_6_valid
## accuracy                      0.9375   0.9354839   0.9587629   0.9162561
## auc                        0.9769779  0.96786684   0.9840872   0.9586375
## err                           0.0625  0.06451613 0.041237112  0.08374384
## err_count                       11.0        12.0         8.0        17.0
## f0point5                  0.95446587   0.9429477  0.96049047  0.92907804
## f1                         0.9519651       0.952   0.9724138   0.9390681
## f2                         0.9494774   0.9612278  0.98463684   0.9492754
## lift_top_group             1.5304347   1.5121951   1.3661972   1.4817518
## logloss                      0.19698  0.20967759  0.16178817  0.25402424
## max_per_class_error       0.08196721  0.12698413  0.13461539  0.16666667
## mcc                        0.8626064   0.8547328  0.89391583  0.80667025
## mean_per_class_accuracy   0.93292946   0.9202478   0.9291712  0.89476883
## mean_per_class_error     0.067070566  0.07975223  0.07082882  0.10523114
```

```
## mse                      0.057315197 0.058498546 0.040672716 0.070079185
## precision                 0.95614034  0.93700784   0.9527027  0.92253524
## r2                         0.7469144   0.7388288  0.79269254  0.68061346
## recall                     0.9478261  0.96747965   0.9929578  0.95620435
## rmse                      0.23940593  0.24186473  0.20167477  0.26472473
## specificity               0.91803277   0.8730159  0.86538464   0.8333333
##                           cv_7_valid  cv_8_valid cv_9_valid cv_10_valid
## accuracy                   0.9126214   0.9453552   0.920904   0.9470588
## auc                        0.9493719   0.9829365  0.9570912    0.985005
## err                       0.08737864  0.05464481 0.07909604 0.052941177
## err_count                       18.0        10.0       14.0         9.0
## f0point5                   0.9317212  0.96746576 0.95158595  0.95284873
## f1                         0.9357143   0.9576271 0.94214875  0.95566505
## f2                        0.93974173   0.9479866  0.9328969    0.958498
## lift_top_group             1.4820144       1.525  1.4390244   1.6831683
## logloss                   0.26470956  0.19371651 0.24186061  0.18362957
## max_per_class_error       0.14925373 0.058333334 0.09259259 0.072463766
## mcc                       0.79957104  0.88176966  0.8183729  0.89004207
## mean_per_class_accuracy   0.89659613   0.9470238  0.9171183  0.94396615
## mean_per_class_error     0.103403844  0.05297619 0.08288166 0.056033865
## mse                       0.07118255 0.055463616 0.06909897 0.051511213
## precision                 0.92907804   0.9741379  0.9579832  0.95098037
## r2                        0.67564666  0.75430936  0.6740738  0.78638625
## recall                    0.94244605  0.94166666  0.9268293  0.96039605
## rmse                      0.26680058  0.23550715 0.26286682  0.22696082
## specificity                0.8507463  0.95238096  0.9074074  0.92753625
##                          cv_11_valid cv_12_valid cv_13_valid cv_14_valid
## accuracy                   0.9494382  0.94711536      0.9375   0.9398907
## auc                       0.98661906    0.975469    0.972929   0.9716619
## err                      0.050561797 0.052884616      0.0625  0.06010929
## err_count                        9.0        11.0        13.0        11.0
## f0point5                   0.9461664   0.9660574  0.96231496   0.9549689
## f1                         0.9626556   0.9641694  0.95652175  0.95719844
## f2                         0.9797297   0.9622887  0.95079786   0.9594384
## lift_top_group             1.5213675   1.3506494   1.3774835   1.4296875
## logloss                   0.18400167  0.19081248  0.19495863  0.21208708
## max_per_class_error       0.13114753  0.09259259   0.0877193  0.10909091
## mcc                        0.8881837  0.86332273  0.84598863  0.85635924
## mean_per_class_accuracy    0.9301527   0.9342232   0.9296503   0.9259233
## mean_per_class_error      0.06984728  0.06577682 0.070349716 0.074076705
## mse                      0.049808357  0.05335354 0.057592176 0.061682552
## precision                  0.9354839  0.96732026   0.9662162  0.95348835
## r2                         0.7788808  0.72242814   0.7105068  0.70657855
## recall                      0.991453  0.96103895   0.9470199   0.9609375
## rmse                      0.22317787  0.23098385  0.23998371  0.24835972
## specificity               0.86885244   0.9074074   0.9122807   0.8909091
##                          cv_15_valid cv_16_valid cv_17_valid cv_18_valid
## accuracy                   0.9221557  0.92941177  0.93406594  0.92899406
## auc                       0.95682627  0.96940106  0.96871305    0.945913
## err                      0.077844314  0.07058824  0.06593407  0.07100592
```

```
## err_count                     13.0        12.0        12.0        12.0
## f0point5                 0.9395425    0.953125   0.9621451  0.93333334
## f1                      0.94650203    0.953125    0.953125  0.95454544
## f2                       0.9535655    0.953125  0.94427246   0.9767442
## lift_top_group           1.3916667    1.328125     1.35625   1.3307086
## logloss                 0.23016892  0.21390663  0.36277843  0.24476203
## max_per_class_error     0.17021276  0.14285715  0.07692308  0.26190478
## mcc                      0.8045505  0.81026787  0.84327406  0.80546314
## mean_per_class_accuracy 0.89406025   0.9051339   0.9307692  0.86511064
## mean_per_class_error   0.105939716 0.094866075  0.06923077   0.1348894
## mse                    0.066072926  0.06254792 0.054539576 0.074973375
## precision               0.93495935    0.953125  0.96825397    0.919708
## r2                      0.67327875   0.6637584   0.7327561   0.5985537
## recall                   0.9583333    0.953125  0.93846154    0.992126
## rmse                    0.25704655   0.2500958   0.2335371  0.27381265
## specificity             0.82978725  0.85714287   0.9230769   0.7380952
##                        cv_19_valid cv_20_valid cv_21_valid cv_22_valid
## accuracy                0.93193716  0.94285715  0.96756756   0.9506173
## auc                      0.9653398   0.9756875  0.98695254  0.98357075
## err                     0.06806283 0.057142857 0.032432433 0.049382716
## err_count                     13.0        12.0         6.0         8.0
## f0point5                 0.9356288   0.9526699  0.97110754  0.95238096
## f1                      0.95057034   0.9631902   0.9758065  0.96666664
## f2                       0.9659969  0.97394544  0.98055106   0.9813875
## lift_top_group           1.4921875      1.3125    1.504065   1.3846154
## logloss                 0.23136492  0.19511554  0.17319117  0.15156467
## max_per_class_error     0.15873016        0.18  0.06451613  0.15555556
## mcc                      0.8446536   0.8385571    0.926913  0.87572414
## mean_per_class_accuracy  0.9089162    0.900625  0.95961183   0.9179487
## mean_per_class_error    0.09108383    0.099375 0.040388145 0.082051285
## mse                     0.06731529 0.055915814  0.04574462   0.0423923
## precision                0.9259259  0.94578314       0.968   0.9430894
## r2                       0.6954701  0.69176406   0.7947011   0.7886907
## recall                   0.9765625     0.98125  0.98373985    0.991453
## rmse                     0.2594519  0.23646525  0.21387991   0.2058939
## specificity             0.84126985        0.82   0.9354839  0.84444445
##                        cv_23_valid cv_24_valid cv_25_valid cv_26_valid
## accuracy                0.91099477   0.9230769  0.91061455  0.90361446
## auc                      0.9696801  0.96802604   0.9642641    0.963211
## err                     0.08900524  0.07692308  0.08938547  0.09638554
## err_count                     17.0        14.0        16.0        16.0
## f0point5                0.91374266  0.95524955   0.9450727   0.9160959
## f1                       0.9363296  0.94067794       0.936  0.93043476
## f2                      0.96006143  0.92654425   0.9270998   0.9452297
## lift_top_group           1.4921875   1.5041323   1.4094489   1.4821428
## logloss                 0.21763203  0.21776639  0.22680107  0.23908634
## max_per_class_error     0.22222222  0.08264463 0.115384616   0.2037037
## mcc                     0.79677093   0.8336803  0.78908676   0.7767154
## mean_per_class_accuracy 0.87717015   0.9258908   0.9029376   0.8758267
## mean_per_class_error    0.12282986   0.0741092  0.09706239  0.12417328
```

```
## mse                      0.06672845 0.065854676  0.06880225 0.073764354
## precision                0.89928055   0.9652174   0.9512195  0.90677965
## r2                       0.69812495   0.7044614  0.66618824   0.6639136
## recall                    0.9765625  0.91735536   0.9212598  0.95535713
## rmse                      0.2583185  0.25662166  0.26230183  0.27159593
## specificity               0.7777778  0.93442625  0.88461536   0.7962963
##                         cv_27_valid cv_28_valid cv_29_valid cv_30_valid
## accuracy                  0.9247312  0.92134833    0.877095   0.9293478
## auc                      0.97310823  0.97550035   0.9536957   0.9667318
## err                      0.07526882  0.07865169  0.12290503  0.07065217
## err_count                      14.0        14.0        22.0        13.0
## f0point5                  0.9332322  0.92654425   0.8768116  0.95689654
## f1                        0.9461538  0.94067794   0.9166667  0.94468087
## f2                        0.9594384  0.95524955  0.96031743   0.9327731
## lift_top_group             1.464567    1.547826   1.4672132   1.5333333
## logloss                  0.21242973   0.2266093  0.25174806   0.2282332
## max_per_class_error      0.16949153  0.15873016  0.36842105       0.075
## mcc                       0.8237921   0.8265832   0.7171776  0.84852815
## mean_per_class_accuracy   0.8995062   0.9032436   0.8116911     0.93125
## mean_per_class_error    0.100493796 0.096756384  0.18830888     0.06875
## mse                      0.06184408  0.06823833  0.08067384 0.065495126
## precision                  0.924812  0.91735536  0.85211265   0.9652174
## r2                        0.7144591  0.70157856   0.6282901  0.71127564
## recall                   0.96850395   0.9652174   0.9918033       0.925
## rmse                      0.2486847   0.2612247  0.28403142  0.25592014
## specificity               0.8305085  0.84126985   0.6315789      0.9375
##                         cv_31_valid cv_32_valid cv_33_valid cv_34_valid
## accuracy                  0.9361702    0.884058   0.9476744   0.9408602
## auc                       0.9765898  0.94048876  0.97451437   0.9846675
## err                      0.06382979  0.11594203  0.05232558 0.059139784
## err_count                      12.0        24.0         9.0        11.0
## f0point5                  0.9635812   0.9009629   0.9461664   0.9548611
## f1                        0.9548872  0.91608393   0.9626556  0.95238096
## f2                        0.9463487   0.9317212   0.9797297   0.9499136
## lift_top_group            1.3925925   1.4892086   1.4700855   1.6034483
## logloss                  0.19529128  0.28786054   0.1945776  0.18610537
## max_per_class_error       0.0754717  0.23529412  0.14545454 0.071428575
## mcc                      0.84699905   0.7320492    0.879655   0.8744277
## mean_per_class_accuracy  0.93263453  0.85357594   0.9229992  0.93842363
## mean_per_class_error    0.067365475  0.14642404 0.077000774 0.061576355
## mse                      0.05589251  0.08739081 0.051522408 0.053453173
## precision                 0.9694657  0.89115644   0.9354839  0.95652175
## r2                       0.72390425  0.60382897  0.76313305  0.77225786
## recall                   0.94074076  0.94244605    0.991453  0.94827586
## rmse                     0.23641597  0.29561937  0.22698548  0.23119943
## specificity               0.9245283   0.7647059   0.8545455   0.9285714
##                         cv_35_valid cv_36_valid cv_37_valid cv_38_valid
## accuracy                   0.920904   0.9483568  0.94634145  0.91935486
## auc                       0.9646547   0.9845929  0.98097885  0.97354823
## err                      0.07909604 0.051643193 0.053658538  0.08064516
```

```
## err_count                        14.0        11.0        11.0        15.0
## f0point5                   0.92621666     0.95625  0.97451276  0.93607306
## f1                              0.944  0.96529967   0.9594096   0.9425287
## f2                          0.9624796   0.9745223   0.9447674   0.9490741
## lift_top_group              1.4123682   1.3653846   1.4748201   1.4418604
## logloss                    0.38864395  0.17515951  0.18410112  0.20031828
## max_per_class_error        0.19642857  0.14035088   0.0647482  0.15789473
## mcc                          0.814676  0.86612517  0.88298833   0.8080528
## mean_per_class_accuracy    0.88938904  0.92020917  0.95247436   0.8977968
## mean_per_class_error       0.11061098  0.07979082 0.047525615  0.10220318
## mse                       0.062465835 0.048032053  0.04989624 0.060116474
## precision                   0.9147287   0.9503106   0.9848485   0.9318182
## r2                          0.7111877   0.7549296  0.77143127  0.71715087
## recall                      0.9752066   0.9807692   0.9352518  0.95348835
## rmse                       0.24993166  0.21916215  0.22337466  0.24518661
## specificity                 0.8035714   0.8596491    0.969697  0.84210527
##                           cv_39_valid cv_40_valid cv_41_valid cv_42_valid
## accuracy                   0.94857144   0.9306931   0.9273743  0.91525424
## auc                          0.984498  0.97203654    0.970088   0.9614826
## err                        0.05142857  0.06930693   0.0726257 0.084745765
## err_count                         9.0        14.0        13.0        15.0
## f0point5                   0.95022625  0.94109195   0.9496753  0.94383776
## f1                          0.9655172   0.9492754  0.94736844  0.94163424
## f2                          0.9813084   0.9576023   0.9450727  0.93944097
## lift_top_group              1.3779528   1.4852941   1.4435484   1.3720931
## logloss                    0.16205977  0.21623117   0.2229956  0.23114163
## max_per_class_error        0.16666667  0.13636364  0.10909091  0.14583333
## mcc                         0.8695085   0.8408522   0.8303283  0.78708297
## mean_per_class_accuracy     0.9127297   0.9134358  0.91722876   0.8960756
## mean_per_class_error       0.08727034  0.08656417 0.082771264 0.103924416
## mse                       0.044648424 0.061807718   0.0644598  0.06923242
## precision                   0.9402985   0.9357143   0.9512195   0.9453125
## r2                         0.77569586   0.7190283   0.6971618  0.64971215
## recall                       0.992126   0.9632353   0.9435484  0.93798447
## rmse                       0.21130173  0.24861158  0.25388935  0.26312053
## specificity                 0.8333333   0.8636364   0.8909091   0.8541667
##                           cv_43_valid cv_44_valid cv_45_valid cv_46_valid
## accuracy                   0.94285715   0.9230769  0.93333334  0.90909094
## auc                         0.9706929   0.9677419  0.96665215   0.9488304
## err                       0.057142857  0.07692308  0.06666667  0.09090909
## err_count                        10.0        14.0        14.0        17.0
## f0point5                   0.96345514   0.9279141  0.93911916  0.91352856
## f1                          0.9586777   0.9453125  0.95394737   0.9390681
## f2                         0.95394737   0.9633758  0.96925133   0.9660767
## lift_top_group              1.4344262    1.467742    1.418919    1.406015
## logloss                    0.21619517  0.21944155  0.20541325  0.24746177
## max_per_class_error         0.0754717  0.18965517  0.17741935   0.2777778
## mcc                         0.8664163  0.82065195   0.8373556  0.77461725
## mean_per_class_accuracy      0.937674  0.89307564   0.9011552  0.85359234
## mean_per_class_error      0.062326014  0.10692436  0.09884481  0.14640768
```

```
## mse                         0.06142077  0.06432131 0.058101133  0.07502667
## precision                   0.96666664   0.9166667  0.92948717  0.89726025
## r2                            0.709092   0.7037571  0.72076505  0.63469684
## recall                       0.9508197   0.9758065   0.9797297   0.9849624
## rmse                        0.24783213  0.25361645  0.24104176  0.27390996
## specificity                  0.9245283   0.8103448  0.82258064   0.7222222
##                            cv_47_valid cv_48_valid cv_49_valid cv_50_valid
## accuracy                     0.9585799   0.9081081   0.9378531    0.920904
## auc                         0.98054224   0.9614172   0.9613361  0.96989566
## err                        0.041420117  0.09189189  0.06214689  0.07909604
## err_count                          7.0        17.0        11.0        14.0
## f0point5                     0.9576271   0.9039548   0.9591195  0.94262296
## f1                          0.96995705  0.93772894  0.95686275  0.94262296
## f2                           0.9826087   0.9741248   0.9546166  0.94262296
## lift_top_group               1.4824561   1.4453125   1.3828125   1.4508197
## logloss                      0.1866811  0.22904293  0.23142505  0.21246149
## max_per_class_error         0.10909091   0.2982456  0.10204082  0.12727273
## mcc                         0.90554786  0.78707033  0.84584093  0.81535023
## mean_per_class_accuracy      0.9410686  0.85087717   0.9255421   0.9076751
## mean_per_class_error       0.058931418   0.1491228 0.074457906  0.09232489
## mse                         0.05021729  0.06960763 0.068232454  0.06374601
## precision                   0.94957983   0.8827586  0.96062994  0.94262296
## r2                            0.771251  0.67347574    0.659175  0.70236975
## recall                      0.99122804         1.0    0.953125  0.94262296
## rmse                        0.22409214  0.26383257  0.26121342  0.25247973
## specificity                  0.8909091   0.7017544   0.8979592   0.8727273
```

```
## H2OBinomialMetrics: drf
##
## MSE:  0.09620613
## RMSE:  0.3101711
## LogLoss:  0.3301547
## Mean Per-Class Error:  0.1197961
## AUC:  0.9554861
## Gini:  0.9109721
##
## Confusion Matrix (vertical: actual; across: predicted) for F1-optimal
threshold:
##         present unknown    Error       Rate
## present    1101     257 0.189249  =257/1358
## unknown      88    1660 0.050343   =88/1748
## Totals     1189    1917 0.111075  =345/3106
##
## Maximum Metrics: Maximum metrics at their respective thresholds
##                         metric threshold    value idx
## 1                       max f1  0.520000 0.905866 212
## 2                       max f2  0.404430 0.942982 243
## 3                 max f0point5  0.680000 0.907154 159
## 4                 max accuracy  0.540000 0.889247 206
## 5                max precision  0.999989 1.000000   0
```

```
## 6                    max recall  0.160000 1.000000 339
## 7               max specificity  0.999989 1.000000   0
## 8              max absolute_mcc  0.520000 0.776006 212
## 9   max min_per_class_accuracy  0.600000 0.874816 186
## 10 max mean_per_class_accuracy  0.540000 0.882215 206
##
## Gains/Lift Table: Extract with `h2o.gainsLift(<model>, <data>)` or
`h2o.gainsLift(<model>, valid=<T/F>, xval=<T/F>)`
```