Using Machine Learning to Determine Who Should Get a Stimulus Check

By Scott L. A. Johnson

Summary

Explored modeling UCI Adult Data Set based on the 1994 Census using Classification models: KNN, Decision Tree, Random Forest, Logistic Regression

Compared performance of models: KNN performed the best

Model citizen who will benefit most from stimulus check: single, younger than 36, didn’t finish high school

Outline

- Business Problem
- Data
- Methods
- Results
- Conclusions

Business Problem

Assumption*: citizens within/under defined income threshold will most benefit from stimulus check

•Want a way to, in the absence a person's immediate income, identify those citizens above* **[develop PERSON PROFILE based on weighted characteristics]**

•Machine learning algorithm can help us weight the most important characteristics associated with a person's income for use under the following example circumstances:

- •Don't have access to recent tax return
- •Citizen's immediate job/financial circumstances have changed
- •Specific industries/communities have been affected by pandemic/natural disaster

Business Problem: Questions

- What factors contribute most to telling us who makes above 50k, least at risk without a stimulus check? What types of people need the most support based on?
- What is the highest precision we can get to minimize the false positives, where we assume a family can make it without a check but can’t?
- How do we minimize the overall cost / impact of the pandemic?
- Is there an industry focus we should have? What industries should we focus on?

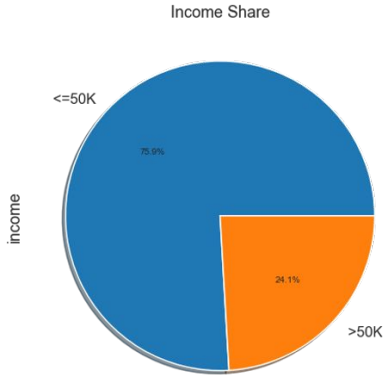
Business Problem: Consequences

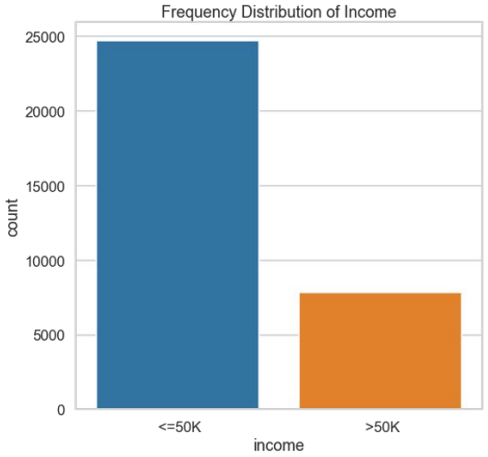
- False Negative: wrong about who makes under 50k
  - Send check [less resources to allocate to others and support future programs]
  - Citizens may use that check frivolously, but contribute back to the economy

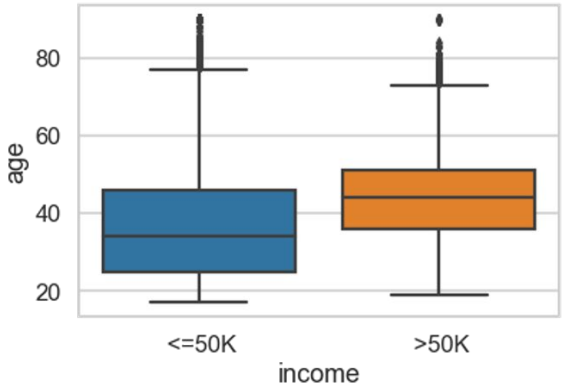
- **False Positive:** wrong about who makes over over 50k
  - No check puts citizens at risk of homeless (cost more), lost job
  - We have more resources to reallocate, but the time lost doesn't help you pay your bills
  - Enact policies to alleviate people who we've missed

Choose the model with the best F1 score

Data ➜ ExploDA

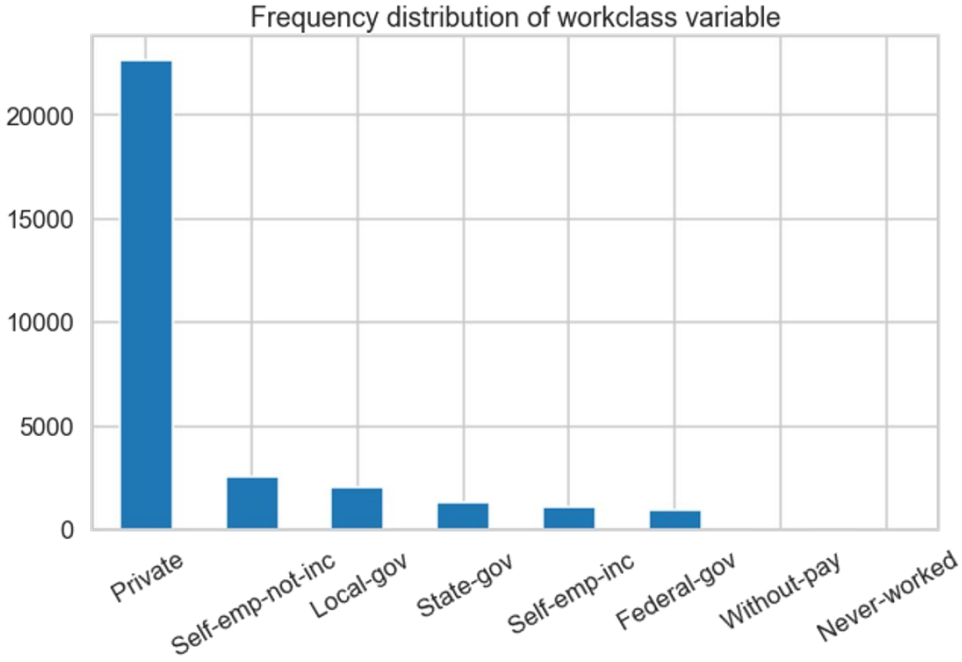






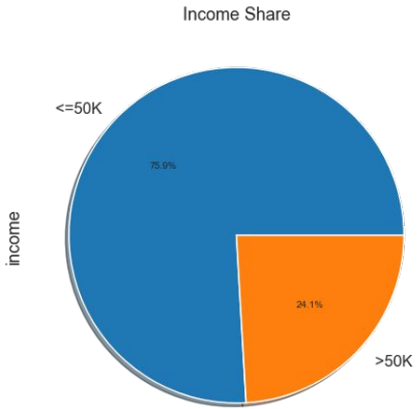
The percentage of the population in dire need of the check coincides with age

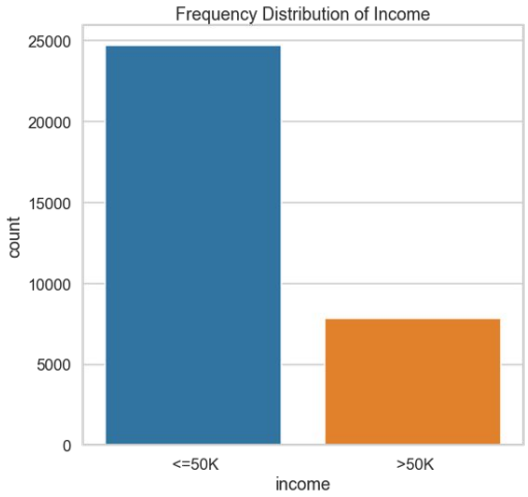
Data ➔ EDA

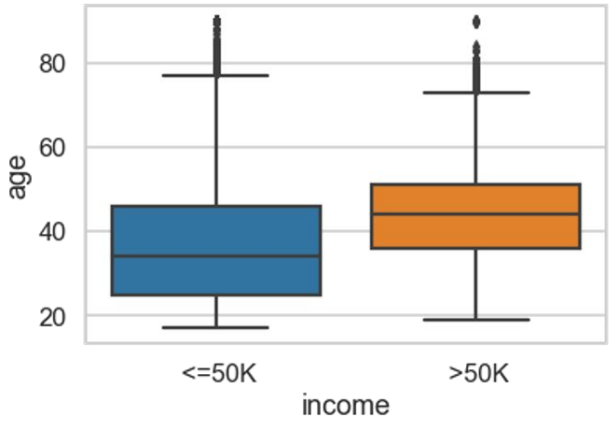


A focus on private industry gives the highest opportunity to address the 75% of the population that makes less than 50K

Data







The percentage of the population in dire need of the check coincides with age

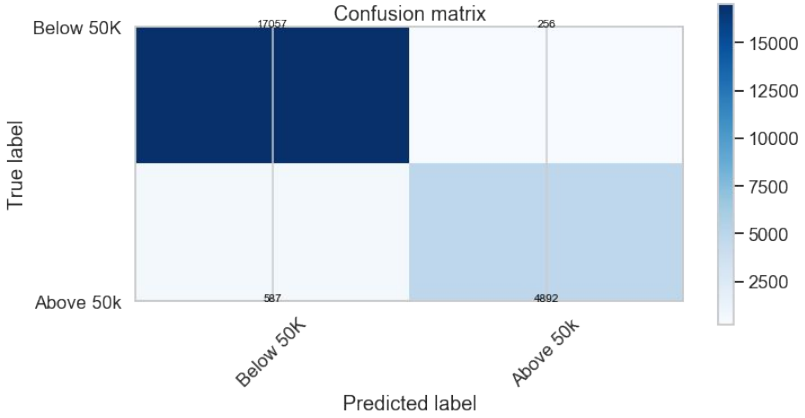
Methods

- Data Cleaning Preprocess
  - Dropped fnlwgt, race, country, sex, education from model
  - Checked for missing data and place holders
- Feature engineering for categorical values
- Shotgun approach to choose a couple models and compare their precision
  - Used SMOTE (**Synthetic Minority Oversampling Technique)** due to the severe class imbalance to improve model performance
- Use feature scoring from models to determine variables with the highest influence on classification

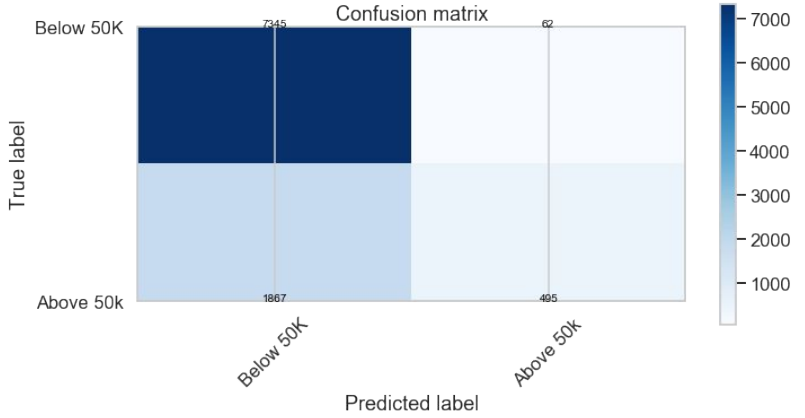
Results (Training Data / Test Data)

Model	Precision	Accuracy	Recall	F1 Score	ROC_AUC
Random Forest **70cv_score:** 0.70 / 0.71	**70:** 0.95 / 0.89	**70:** 0.96 / 0.80	**70:** 0.89 / 0.21	**70:** 0.92 / 0.34	**70:** 0.94 / 0.60

70/30 Train-Test Split: Training Data



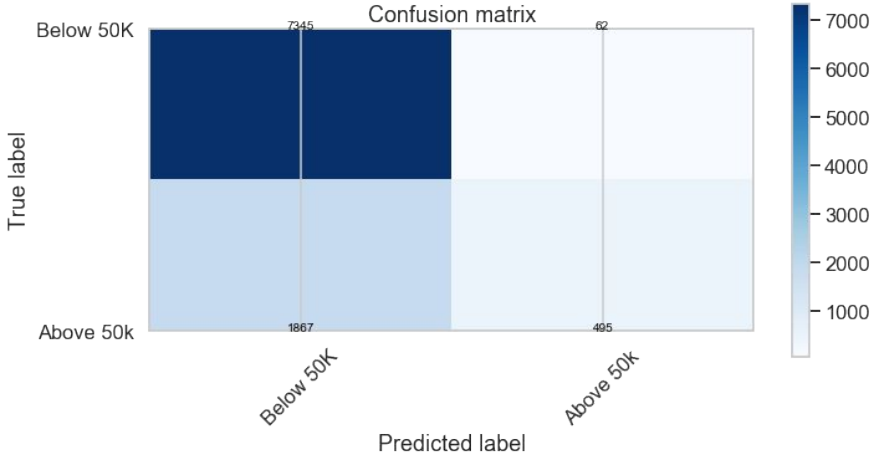
70/30 Train-Test Split: Test Data



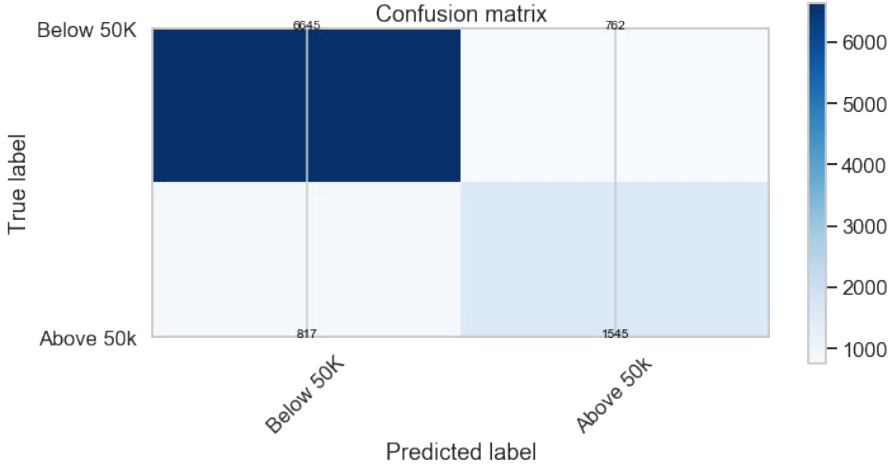
Results (Test Data)

Model	Precision	Accuracy	Recall	F1 Score	ROC_AUC
Random Forest	**70:** 0.89 **70_smt:** 0.67	**70:** 0.80 **70_smt:** 0.84	**70:** 0.21 **70_smt:** 0.65	**70:** 0.34 **70_smt:** 0.66	**70:** 0.60 **70_smt:** 0.78

70/30 Train-Test Split: Test



70/30 Train-Test Split: Test w/ SMOTE



Results (Test Data)

Model	Precision	Accuracy	Recall	F1 Score	ROC_AUC
Random Forest (cv_score =0.70)	**70:** 0.93 **70_smt:** 0.67 **80:** 0.92 **80_smt:** 0.66	**70:** 0.79 **70_smt:** 0.84 **80:** 0.79 **80_smt:** 0.84	**70:** 0.16 **70_smt:** 0.65 **80:** 0.18 **80_smt:** 0.66	**70:** 0.27 **70_smt:** 0.66 **80:** 0.30 **80_smt:** 0.66	**70:** 0.57 **70_smt:** 0.78 **80:** 0.59 **80_smt:** 0.78
KNN [15] (cv_score =0.69)	**70:** 0.79 **70_smt:** 0.67 **80:** 0.77 **80_smt:** 0.66	**70:** 0.86 **70_smt:** 0.85 **80:** 0.85 **80_smt:** 0.84	**70:** 0.55 **70_smt:** 0.72 **80:** 0.55 **80_smt:** 0.73	**70:** 0.65 **70_smt:** 0.70 **80:** 0.65 **80_smt:** 0.70	**70:** 0.75 **70_smt:** 0.80 **80:** 0.75 **80_smt:** 0.80
Decision Tree [11] (cv_score =0.71)	**70_smt:** 0.58 **80_smt:** 0.58	**70_smt:** 0.81 **80_smt:** 0.81	**70_smt:** 0.78 **80_smt:** 0.79	**70_smt:** 0.67 **80_smt:** 0.67	**70_smt:** 0.80 **80_smt:** 0.80
LogReg (cv_score = 0.73)	**70_smt:** 0.65 **80_smt:** 0.64	**70_smt:** 0.84 **80_smt:** 0.83	**70_smt:** 0.71 **80_smt:** 0.71	**70_smt:** 0.68 **80_smt:** 0.67	**70_smt:** 0.80 **80_smt:** 0.58

Conclusions

Our best performing model was the KNN model with a CV score of .70 , a F1 score of .70,

Candidate income profile heavily influenced from our random forest model:

- Age ➜ focus on citizens < 36
- Marital Status ➜ focus on single
- Education ➜ focus on education < 10th grade (those who don't have a high school diploma)

*Workclass/occupation did not rank high for the 1994 data set as an income determinant

- Does not provide accurate representation of factors that would be relevant to today's job market

Future Recommendations

- Feed next rounds of data into model to train, collect information relevant to industries of relevant today -->clearer
- Bin wage classes differently to assign different stimulus amounts based on wage classes
- Alternate analysis can also be done with respect to wage class ➔ different factors might distinguish those in the 150k and up class vs 75k-90k ➔ zipcode could segment the data ➔ identify risk of being able to weather pandemic (vaccine selection)
- Though not done in this study might be interesting to explore wage disparities by community / culture ➔ if funds are dispense at a state level, could prioritize support based on tiered approach [zipcode]

Thank You!

Email: scottlajohnson92@gmail.com
GitHub: @scottlou
LinkedIn: www.linkedin.com/in/scott-johnson-432b83163/