Introduction

For our project, we plan to use census data containing information such as age, education level, sex, marital status, capital gain/loss that year, etc. to predict whether a person is likely to have an income of over $50k or not. This dataset is obtained from the UCI Machine Learning repository, so we feel that it could give us more easily interpretable results as compared to real data.

Objective and Business Questions

How great would it be for local businesses to tailor services to their customers' needs? Given that a lot of census data is public, companies can use data from these to drive their business decisions – for example, if there is a large population from a certain country, selling food originating from those countries could be a worthwhile endeavor. Being able to predict whether one's customers fall in the high or low-income categories can have a significant impact on what type of goods one would stock up in their shop. Another business question which could be answered is how much the age of a population affects income. If a lot of people in their 20s are in the lower income group, then having expensive, say gaming equipment, wouldn't make sense.

Data Description

The income dataset has about 48k entries and is obtained from https://archive.ics.uci.edu/ml/datasets/census+income. Income is the categorical variable we try to predict as " <=50K." or " >50K.". There is an approximately 1:3 split between over $50K and under $50K. The dataset is also split to train and test sets with approximately 32k and 16k entries respectively.

The following table represents a summary of variables, description and examples:

Variable	Type	Example
Age	Continuous	28, 40
Workclass	Categorical	Private, Self-emp
Final Weight	Continuous	228654
Education	Categorical	Doctorate, HS-Grad
Education Level	Continuous	15, 9
Marital Status	Categorical	Divorced, Married
Occupation	Categorical	Tech-support, Exec-manager

Variable	Type	Example
Relationship	Categorical	Wife, Unmarried
Race	Categorical	Asian-Pac-Islander, Other
Sex	Categorical	Male, Female
Capital Gain	Continuous	51478
Capital Loss	Continuous	7843
Hours per week	Continuous	40
Native Country	Categorical	United States, Holland-Netehrlands
Income	Categorical	>50K, <=50K

Data Preprocessing and Preparation

For pre-processing, we had to remove the junk row at the top of test set. This was done manually in the csv, before converting into arff for Weka for Naïve Bayes and Random Forest. For use in Python, we took this data directly from the UCI repository, so we removed it in Databricks.

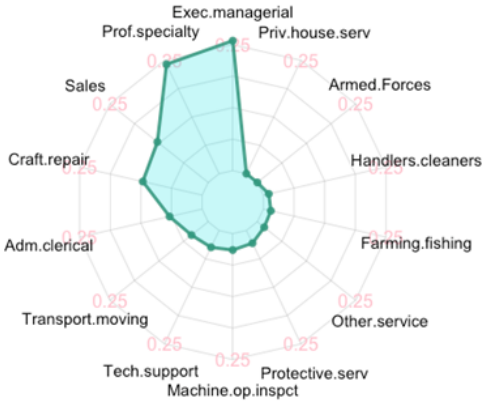
There was also a lot of unknown data shown as "?" in the dataset in "Work Class", "Occupation" and "Native-Country". These are rather small in number (under our threshold of 7.5% of the entire data), so we decided to remove them from the dataset.

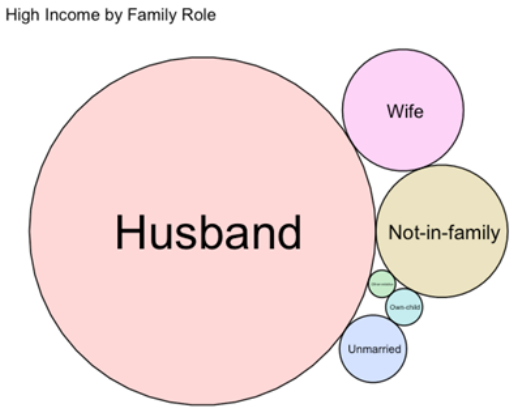
Furthermore, there are many categorical variables – like Native-Country, Occupation, etc. For logistic regression in Python, we need to make (n-1) dummy variables for each categorical column and cast into "int" type.

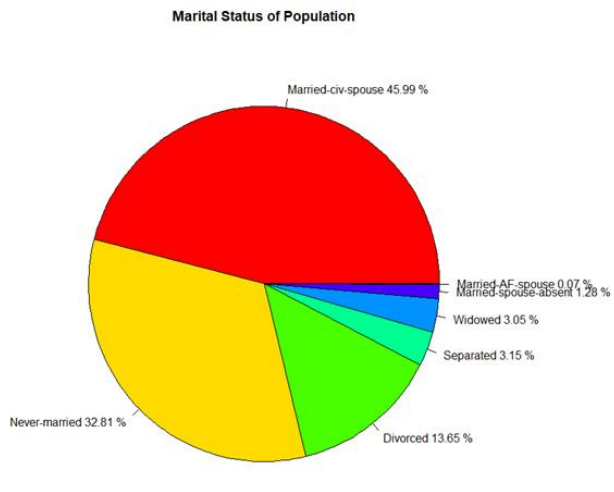
In Python, we split the dataset into three – one for training, one for validating each model, and one for testing – in a ratio of 46.666: 20: 33.333.

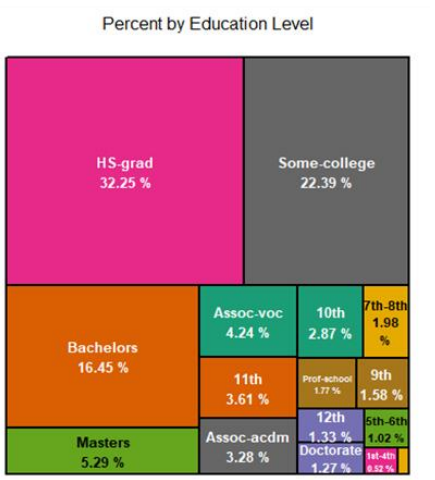
Data Analysis

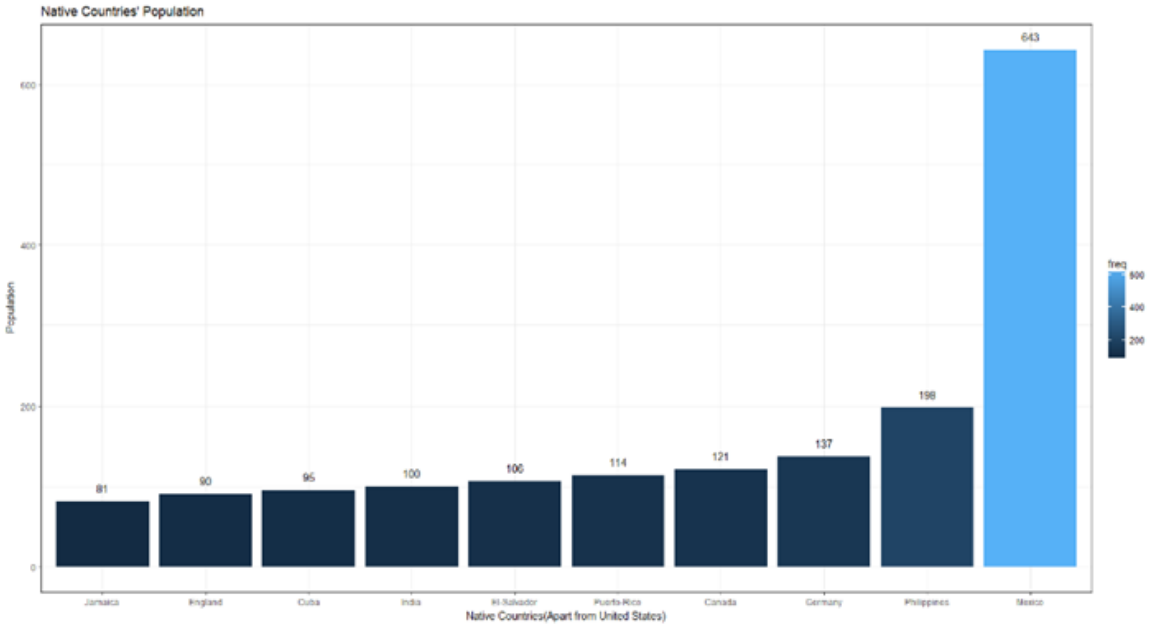
The following are some descriptive statistics from the dataset:









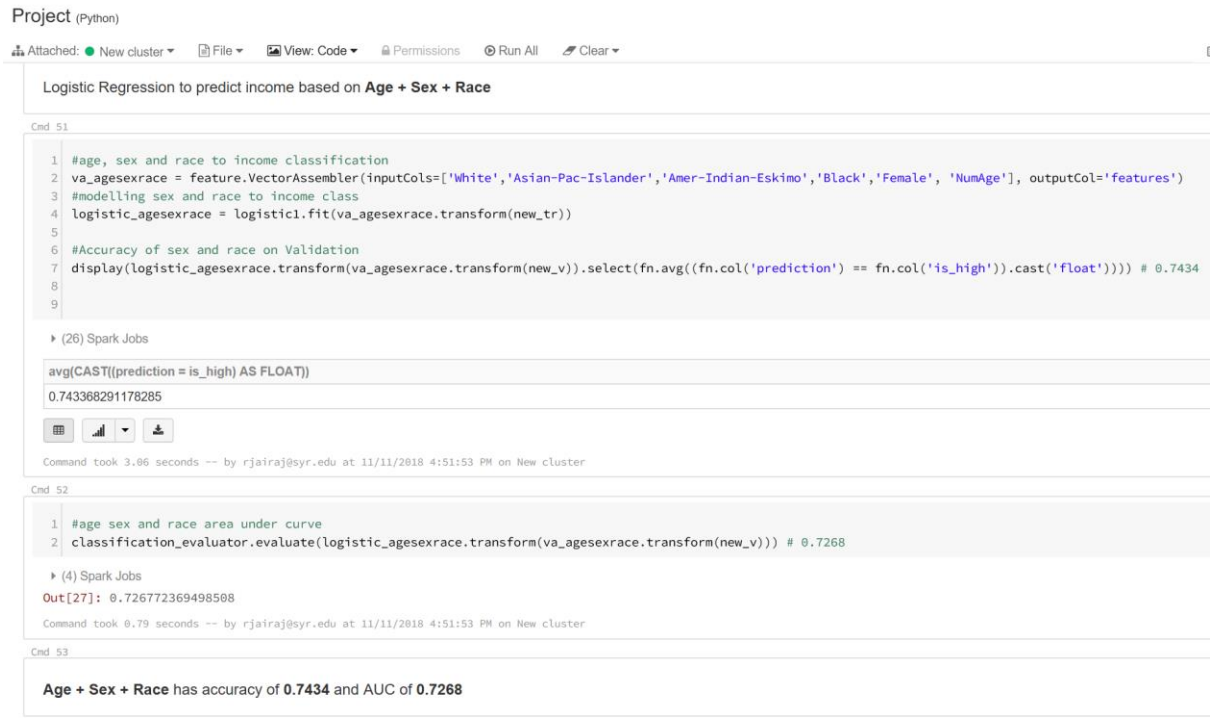


Logistic Regression

For the initial processing, we tried to think logically about what factors could affect the income of a person. We came up with the following 16 possibilities to work with initially:

Education	Sex + Race + Marital Status	Workclass	Education + Occupation
Age	Age + Sex + Race	Age + Sex	Education + Occupation + Workclass
Sex	Age + Sex + Race + Country	Marital Status + Sex	Education + Occupation + Workclass + Age + Sex + Marital Status
Occupation	Country + Race	Age + Race	All features

We used the following code to measure the accuracy and area under the curves:



We obtained the following results:

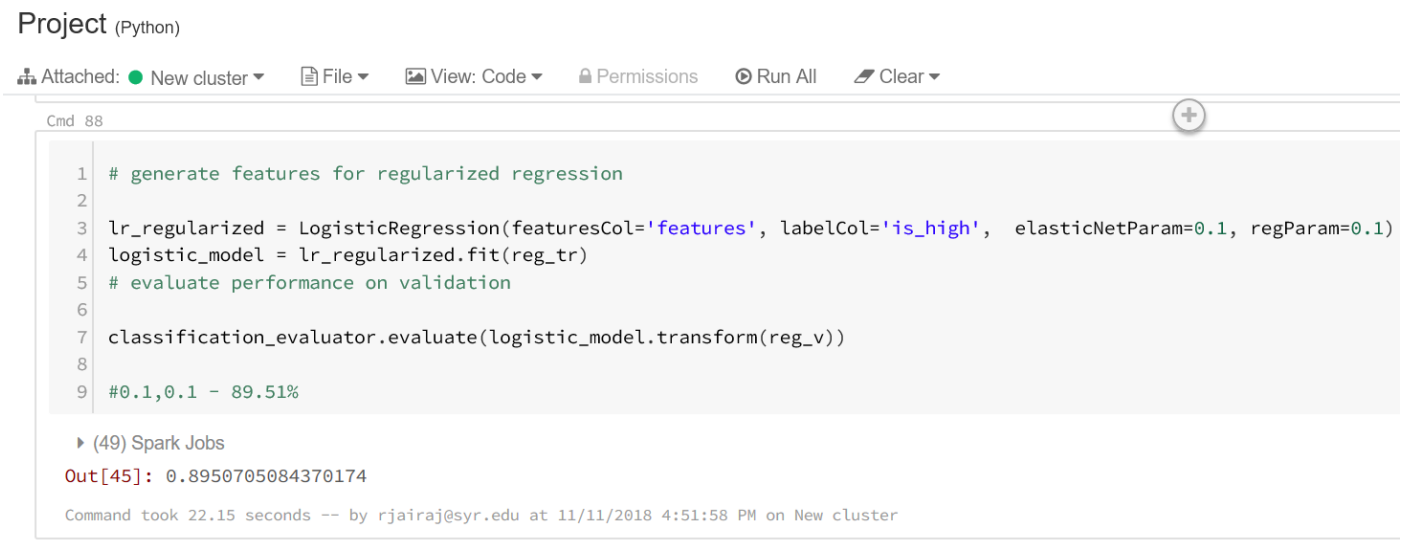
Features	Accuracy	Area Under Curve
Education Number	0.7797	0.7172
Age	0.6837	0.6208
Sex	0.7608	0.5388
Occupation	0.7608	0.7265
Workclass	0.7652	0.5928
Age + Sex	0.7413	0.7241
Marital Status + Sex + Age	0.7472	0.7992
Age + Race	0.7428	0.691

Features	Accuracy	Area Under Curve
Sex + Race	0.7608	0.638
Age + Sex + Race	0.7434	0.7268
Age + Sex + Race + Country	0.7435	0.7322
County + Race	0.7608	0.5568
Education + Occupation	0.7814	0.7675
Education + Occupation + Workclass	0.7851	0.7742
Education + Occupation + Workclass + Age + Sex + Marital Status	0.8356	0.8814
All features	0.8536	0.9052

As can be observed, we obtained the best results when we use all the features, with an accuracy of **85.36%** and area under the curve of **90.52%.** The interpretation is that using **all the attributes** helps us predict the income of a person the **best**.

Regularized Logistic Regression

Next, we try regularized logistic regression, varying the values of α and λ.



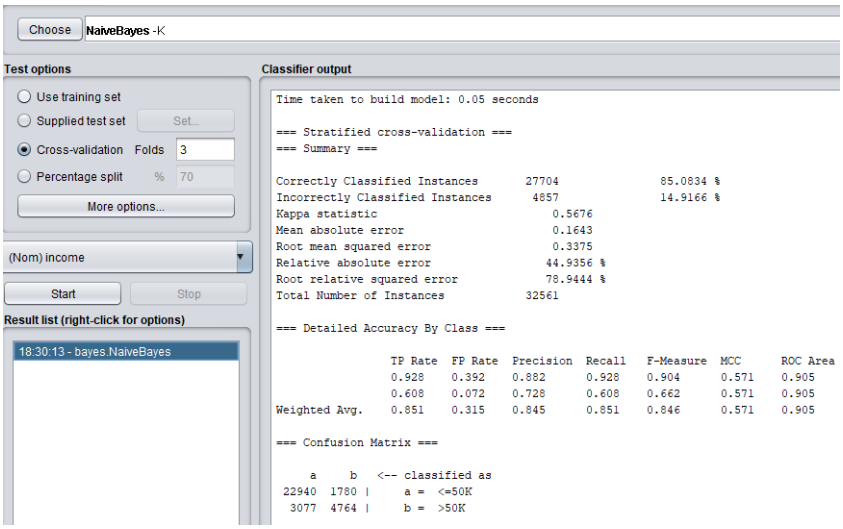
Judging from the area under the curve, we find that with $\alpha = \lambda = 0$, we get the best result.

Alpha	Lambda	Area Under Curve
0	0	0.9052
0.05	0.05	0.8991
0.3	0.05	0.894
0.1	0.1	0.8951
0.05	0.3	0.8893
0.2	0.2	0.8826
0.3	0.3	0.8533

The implication is that ordinary logistic regression would be a better choice, than this.

Naïve Bayes

For Naïve Bayes, we try to classify using kernel estimator, supervised discretization and default setting with 3-fold cross validation to prevent over-fitting.

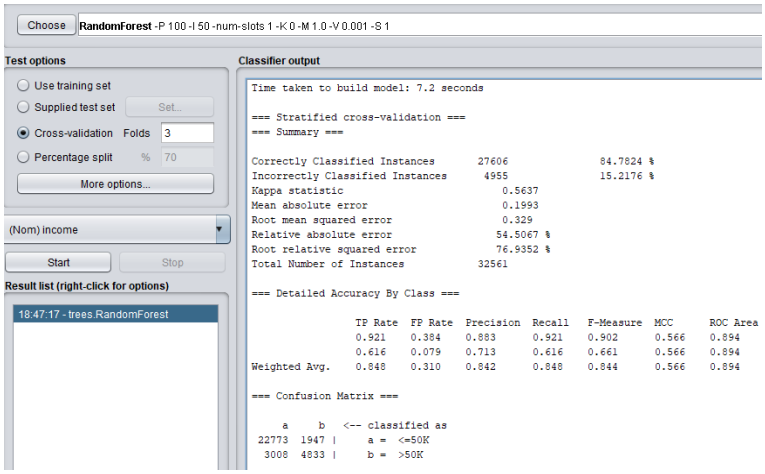


Type	Accuracy	Area Under Curve
Default	0.8339	0.892
Supervised Discretization	0.8324	0.912
Kernel Estimator	**0.8508**	**0.905**

So, for Naïve Bayes, the best option is to use kernel estimator. We get accuracy of **85.08%**, with area under curve as **90.5%.** This implies that there is a **high level of independence** amongst the different variables – like Native-Country, Age, Occupation, etc. and this allows us to predict income with a high level of accuracy.

Random Forest

For Random Forest, we try using 25, 50 and 100 trees, again with 3-fold CV.



Number of Trees	Accuracy	Area Under Curve
25	0.8452	0.887
50	**0.8478**	**0.894**
100	0.8475	0.896

The best choice seems to be using 50 trees, with an accuracy of **84.78%** and area under the curve to be **89.4%**. Varying tree depth to 20 showed a decrease in accuracy to **84.77%**. So, using just 50 trees seems to be the best choice in Random Forest.

Association Rules

From the A-Rules' side, we get the following interesting rules:

```
    lhs                                                                  rhs            support    confidence lift     count
[1] {marital-status=Never-married}                                    => {income=<=50K} 0.3068762 0.9516759  1.267081 9256
[2] {age=15-30}                                                       => {income=<=50K} 0.2962668 0.9311243  1.239718 8936
[3] {workclass=Private,sex=Female}                                    => {income=<=50K} 0.2294609 0.9056530  1.205805 6921
[4] {relationship=Not-in-family,capital-gain=0-20000,capital-loss=0-1000} => {income=<=50K} 0.2206419 0.9033528  1.202742 6655
[5] {sex=Female,hours-per-week=20-40}                                 => {income=<=50K} 0.2046615 0.9012995  1.200009 6173
```

This implies that lower age group people, never married people, women working in the private sector and women working between 20 and 40 hours a week all likely fall under the low-income category.

Challenges

The primary challenge was learning Python and syntax errors in Python. Since we realized that the learning curve might be steep, we started well in advance.

Converting all the categorical variables manually to dummy variables was a tedious task. During the mid-point check-in, it was brought to our attention that this could be done using a function in a specific library. However, since it was already done we didn't explore this further. Furthermore, the dummy variable was simply giving an error when used in regression. It took a long while to try casting to "int", which ended up working at the end.

Figuring out how to find standardized weights also took a long time. The code we obtained didn't translate very well to our project.

Conclusion

As we can see from the models created, the best choice is logistic regression. So, we perform testing on this and obtain an accuracy of **85.02%** and area under the curve is **90.22%.**

Let's look into which features have the most weight in predicting income.

Hence, analyzing features after standardization, contributing the most to classification, we find the following attributes have the most weight:

Contributing most to over 50k

	coefficients	weight
73	capital-gain	2.323772
8	education-num	0.716232
18	Exec-managerial	0.538102
19	Prof-specialty	0.466453
79	Not-in-family	0.419473
76	Wife	0.378415
75	hours-per-week	0.369124
0	NumAge	0.353997
17	Sales	0.352427
16	Craft-repair	0.300112

Contributing most to under 50k

	coefficients	weight
11	Never-married	-1.461515
10	Divorced	-0.908309
12	Separated	-0.483412
13	Widowed	-0.434624
72	Female	-0.406331
14	Married-spouse-absent	-0.302044
9	Married-civ-spouse	-0.235444
25	Priv-house-serv	-0.221591
7	Without-pay	-0.206479
37	Outlying-US(Guam-USVI-etc)	-0.182549

This shows that some important factors in deciding whether a person has over 50k income are how much other capital gain they have, their education level, the number of hours they work, and their age. From an occupation standpoint, Exec-Managerial, Prof-Specialty, Sales and Craft-Repair have strong relations to high income. It's also very interesting that Wives, and people who are not in a family are more likely to be in the >50k bracket.

On the flip side, Never-married, divorced, separated, widowed, married-spouse-absent individuals tend to have a lower probability of high income than others. Another interesting point is that people from outlying US territories tend to be of the lower income category. This may be because of very few data points (14) in the training set. It's also unfortunate to note that being a female makes it much more likely that you'd be in the lower income category.

We decided to go with Logistic Regression for this project, as it had the best performance, but the other methods seen also had very nearly the same accuracy - within 1-2% of each other. In conclusion, we'd like to mention that we tried to use many models taught in class in addition to regression, but this project showed us that there could be plenty of ways to successfully perform a classification task.