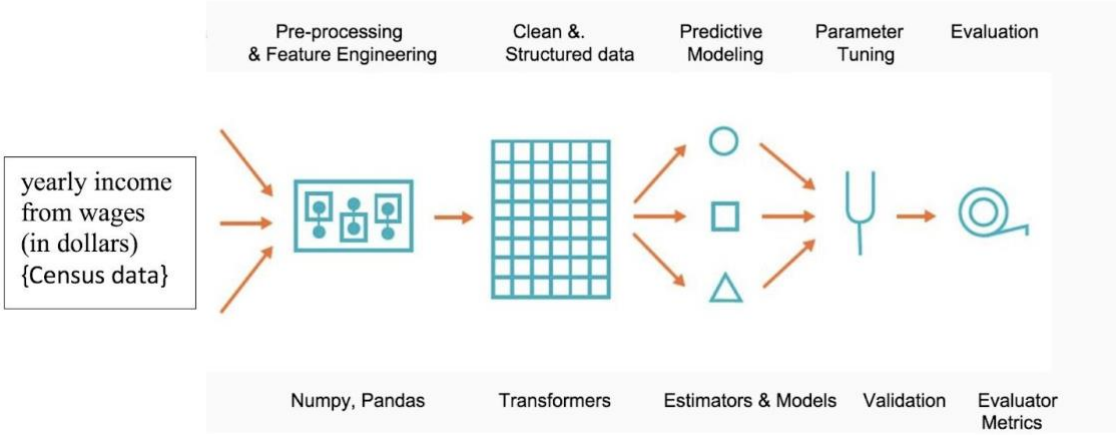
ML Assignment 5: The Income Prediction Problem

Raj Mehta (rcm445)

Problem Statement: The task is to predict the yearly income from wages (in dollars) of a person in the United States, based on other features of the person.

This is a regression problem where we are provided with two datasets (train and test) to predict the yearly income from wages. The Machine Learning algorithms such as linear regression, logistic regression, neural network, SVM, random forests, etc can be used to train the dataset
We decided to work on two algorithms: Logistic regression and Random Forest.

Approach:



Initially we read the census_train and census_test csv files
From the census_train.csv file, we seperate the features and prediction label (in this case wages)

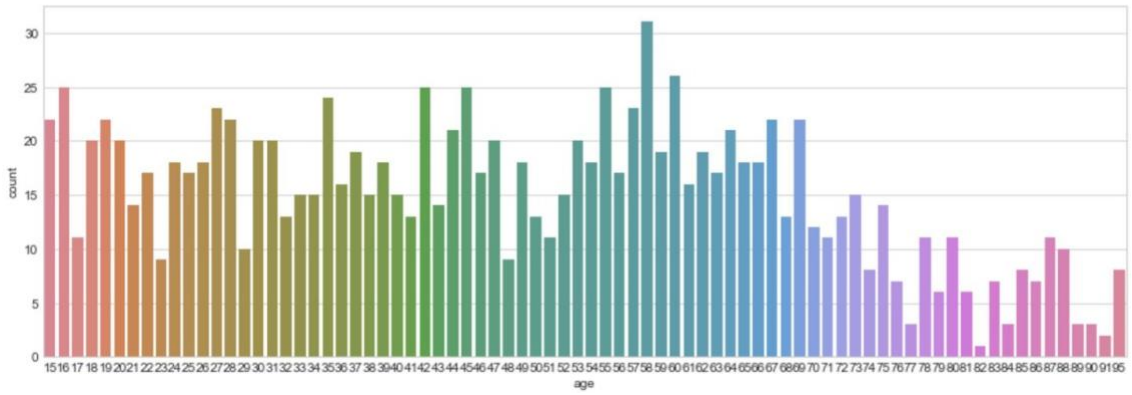
Data Pre-Processing

Often in real-world datasets, there are missing values for some of the features in some of the training and the testing examples. This may be because the feature values for those examples were unavailable or were not recorded. Sometimes the symbol ? is used to indicate a feature that is missing for any reason. In this dataset, the symbol ? is used to indicate the value N/A (not applicable).
Furthermore, we are binning the features “age”, “education_attained” and “industry_worked_in” as the values for these features is spread across a larger range this not helping the model in any

way to make some sense out of those values. Binning overcomes this problem in these features as it helps us to properly distribute the data across intervals in a range which is big enough to be able to distinguish the values in the dataset properly but small enough for it to be redundant.

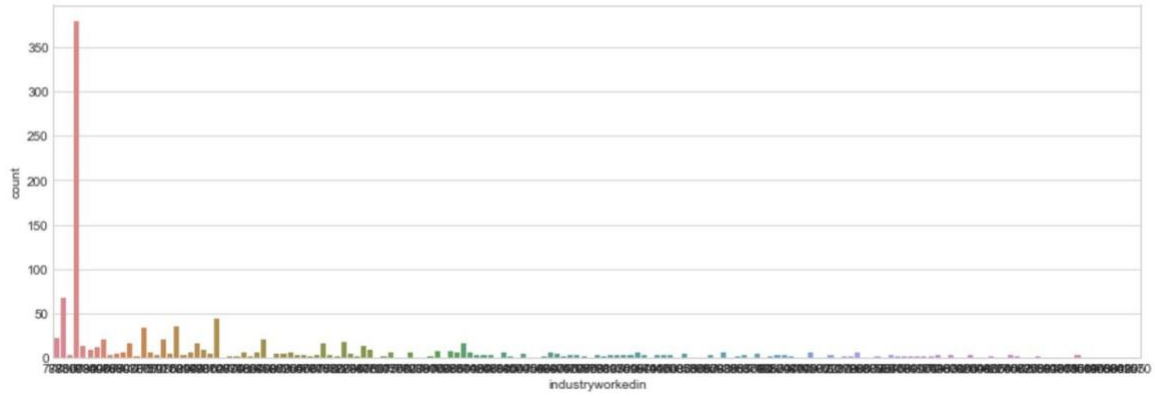
```
fig, (axis1) = plt.subplots(1,1,figsize=(15,5))
sns.countplot(x='age', data=trainData, palette="husl", ax=axis1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a1d90dfd0>
```




```
fig, (axis1) = plt.subplots(1,1,figsize=(15,5))
sns.countplot(x='industryworkedin', data=trainData, palette="husl", ax=axis1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a1ccf3080>
```



It must be noted that the binning in the industry worked in has been carried out by splitting the dataset across industries. Say something like this:

0170 AGR-CROP PRODUCTION
0180 AGR-ANIMAL PRODUCTION AND AQUACULTURE
0190 AGR-FORESTRY EXCEPT LOGGING
0270 AGR-LOGGING
0280 AGR-FISHING, HUNTING AND TRAPPING

0290 AGR-SUPPORT ACTIVITIES FOR AGRICULTURE AND FORESTRY
0370 EXT-OIL AND GAS EXTRACTION
0380 EXT-COAL MINING
0390 EXT-METAL ORE MINING
0470 EXT-NONMETALLIC MINERAL MINING AND QUARRYING
0490 EXT-SUPPORT ACTIVITIES FOR MINING
0570 UTL-ELECTRIC POWER GENERATION, TRANSMISSION AND DISTRIBUTION
0580 UTL-NATURAL GAS DISTRIBUTION
0590 UTL-ELECTRIC AND GAS, AND OTHER COMBINATIONS
0670 UTL-WATER, STEAM, AIR CONDITIONING, AND IRRIGATION SYSTEMS
0680 UTL-SEWAGE TREATMENT FACILITIES
0690 UTL-NOT SPECIFIED UTILITIES
0770 CON-CONSTRUCTION, INCL CLEANING DURING AND IMM AFTER
1070 MFG-ANIMAL FOOD, GRAIN AND OILSEED MILLING
1080 MFG-SUGAR AND CONFECTIONERY PRODUCTS
1090 MFG-FRUIT AND VEGETABLE PRESERVING AND SPECIALTY FOODS

. . . . . .
. . . . . .

Lastly, we check for any duplicates in the dataset and handle it accordingly.

Feature Engineering

1. **Features Dropped**
   Also, while predicting the income wages, it is not mandatoryto use all the features given in the dataset, as some features might not be useful in predicting the outcome (their presence doesn't have any advantage in our prediction). We used feature_importance from RandomForestRegressor module in scikit to help identify some of these irrelevant features. Say for our project, ['idnum' , 'traveltimetowork' , 'vehicleoccupancy' , 'marital', 'ancestry'] are the unnecessary features which do not contribute in our income prediction problem.

   Idnum- its just an index, not a feature

   Traveltimetowork - The time one takes to travel should not be a crucial factor in estimating the income

   Vehicleoccupancy- it doesnt matter if you travel alone or in a group of 2 or in a group of 10. Travelling alone doesnt necessarily mean you might be rich enough to afford it and

hence leading to higher income. Also, travelling in a group also doesnt mean you chip in because of lesser income.

Marital- if you are divorced, or single or married, this should not affect your capabilities in any way, thereby shouldn't affecting the wages

Ancestry- it doesn't matter what your ancestry is Income prediction should not be dependant on features like ancestry

2. **Feature transformations**
   All the categorical feature were converted into multiple features using one-hot-encoding. In this method, for each possible value v of an attribute, we create a new binary-valued attribute whose value is 1 if the original attribute equals v, and 0 otherwise. We use these new attributes in place of the original attribute.
   For our project, the features which undergone one-hot-encoding are:
   "Workerclass", "meansoftransport", "schoolenrollment", "educationalattain", "sex", "degreefield", "industryworkedin"
   Apart from that a lot of string to int conversions were made to feed the machine learning models clean pre-processed data

3. **Feature Scaling**
   We normalized our features by performing Principal Component Analysis as we observed that a lot of our data is skewed. Thus to reduce the feature dimensions so that the machine learning model isn't overwhelmed with the magnitude of the features, we have scaled our features.

Model Selection and Evaluation:

Since the problem was a regression one, we decided to try fitting a linear regression model to the data. We did not expect the model to perform very well, and this was evident in the MSEs that we received. A more promising candidate for the regression seemed a higher degree polynomial regression model. Since the target variable was wages, it was likely to be non-linear. On trying out the model first for degrees 2 and 3 we found that the degree 3 model took very long to train, and required more computing resources than we had.
The ElasticNet model is a combination of Lasso and Ridge Regressions, and is thus a combination of L1 and L2 regularizations. We trained the regression model for degrees 2 and 3 on alpha values ranging from 0.01 to 0.4, and found that the second degree polynomial with a penalizing parameter of 0.2 produced the lowest MSE. A degree 3 polynomial did give a slightly lower MSE, but given the 10x training time that it required over the degree 2, we did not find it worth taking up. Given that the data must be so heavily scattered, we felt that unit changes in the

polynomial degree would not affect the accuracy of the model too much. The regularization value of 0.2 is again a moderate number, and since ElasticNet applies alpha over the L2 norm and (1-alpha) over the L1 norm, the value seems fair to facilitate having a balanced combination of both to update the model.

It was a big improvement over Linear Regression, which gave an average training and test error of 56408.46 and 91449.19 respectively. The second degree polynomial model fit using ElasticNet regression with alpha value as 0.2 gave the best MSE for 5-fold training and test: 63707.26 and 30575.11.

To summarize, the dataset was cleaned, reduced, and trained using second degree polynomial features on an ElasticNet regression model with a regularization parameter of 0.2. The output of the model on the test dataset is as in the Output.csv file attached with the submission.

Also, we tried Random Forest Regressor which performed better than Linear Regression but not better than ElasticNet model. Similar to the steps in above model, we split the data set into the 70% train and 30% validation set, and used the this training set while training the Random Forest Regressor model implemented in Scikit Learn. The results on our 30% validation set are: RMSE: 39628.08