HomeWork 5

Rishi – rr3261

Anoop- adn322

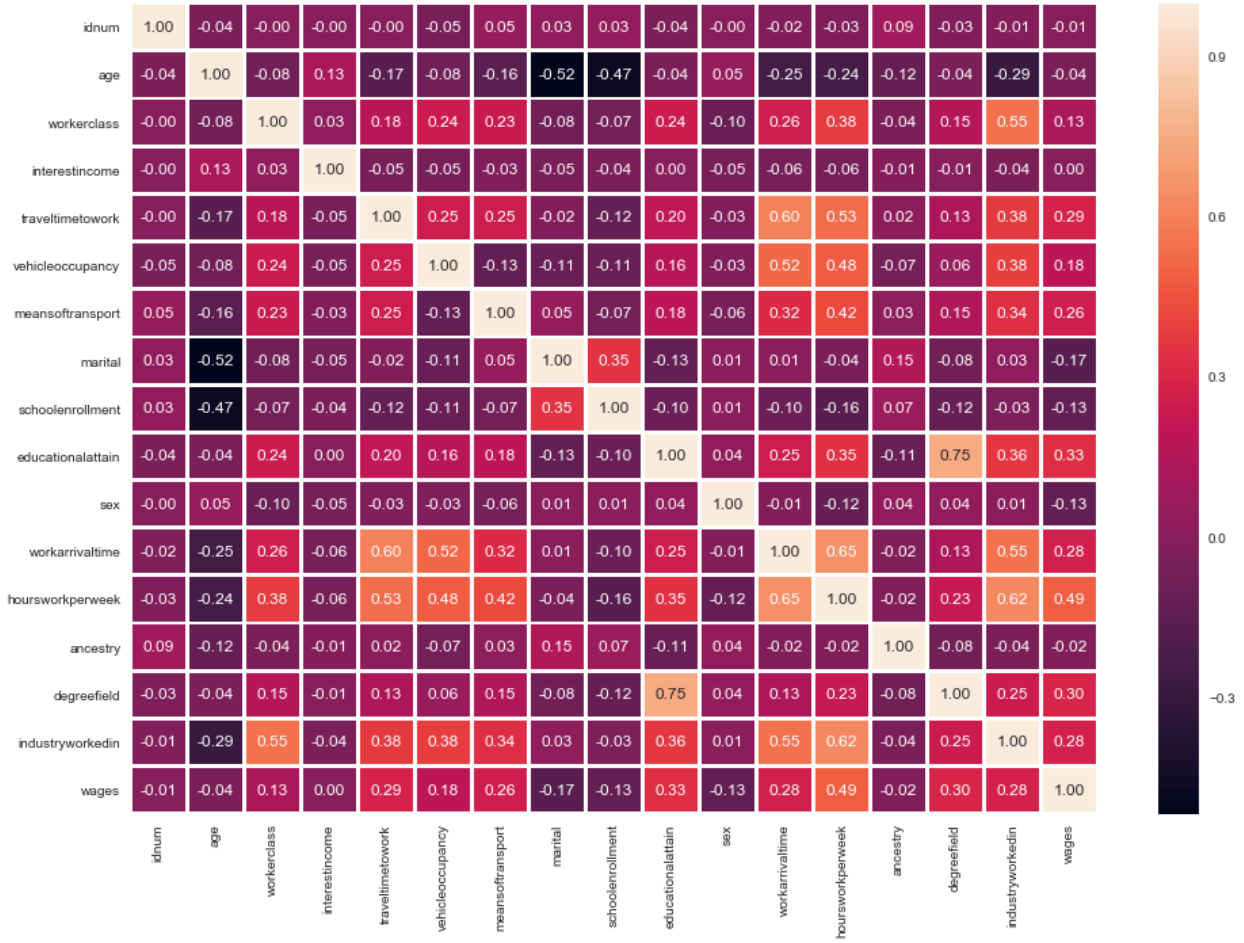
REPORT

Data Cleaning and Preprocessing:

The clearest way to see if there is any relationship amongst the numerical features, categorical features and the wages, is to create a correlation plot. However, due to various categorical features (workerclass, marital, vechicleoccupancy, etc.) it isn't possible to add features not having numerical values directly to the correlation plot. Therefore, I made changes to the categorical features by replacing the '?' with 0 as another category and changing the datatype to numeric.

```
trainfile = trainfile.replace("?",0)
for i in trainfile.columns:
    trainfile[i] = pd.to_numeric(trainfile[i])
```

Once this was fixed a correlation matrix could be easily plotted.

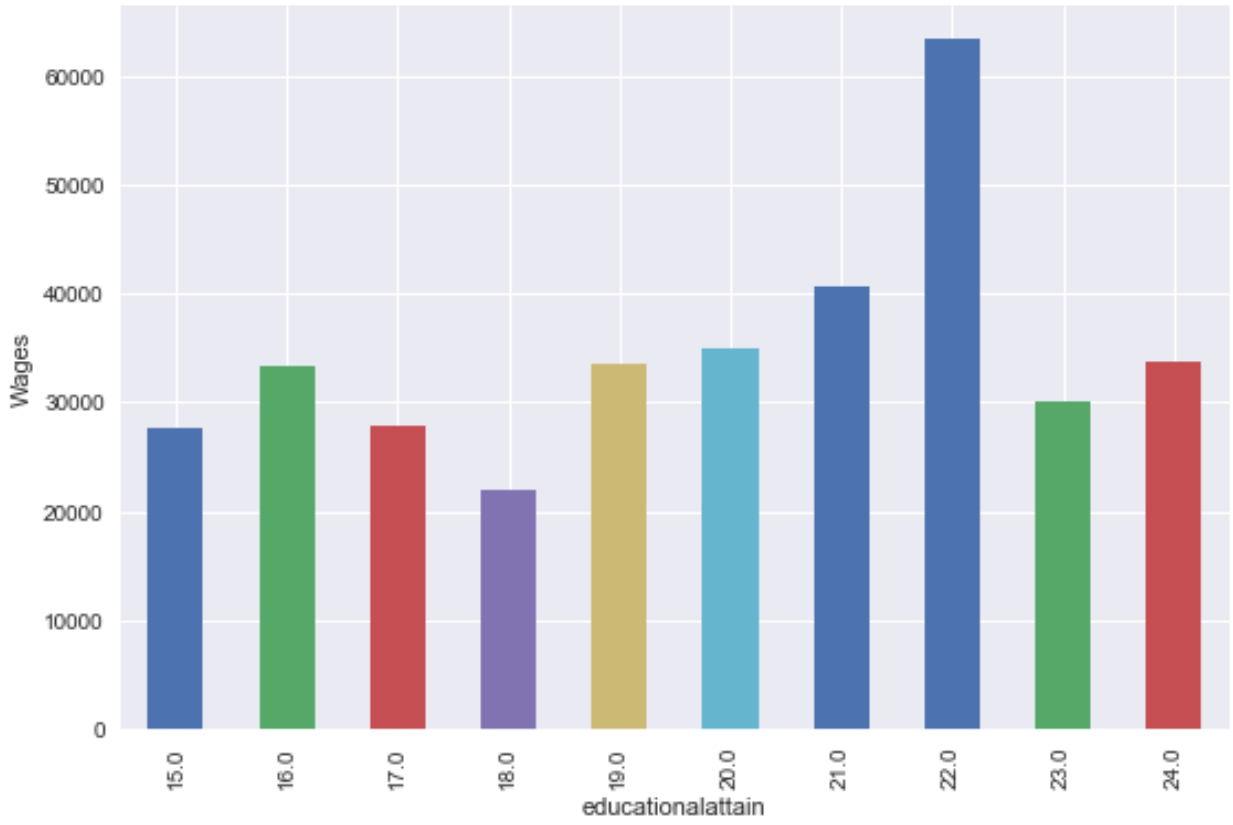


From this Matrix it could easily be seen that hoursperweek, degreefield, educationobtain are some of the most important features and features such as sex, ancestry, age, interestincome doesn't really offer anything.

Hence, we remove the following features:

'age','interestincome','marital','schoolenrollment','sex','ancestry','wages'

One of the most important feature Education attained plot shows how the wages differ by different level of education.



Once the Extraneous features has been removed, we try to simplify the categorical attributes having many features (workarrivaltime, degreefield, industryworkedin) by binning them into similar slots.

Workarrival time bins: Based on shifts we have divided the bin

So 12am-7am : BIN 1
7am-9:30am : BIN 2 and so on…….

For Degreefield and industryworked in we noticed that the similar fields are in the same hundred values i.e. values b/w 2200 and 2299 belong to same field. So we divided this feature into bins of 100.

Scaling Data

Since the value of the category in some sense, **establish a priori assumption** between the features (not really reflection of data itself). And also, the nature of most algorithms is to find the most appropriate weight percentage between the features to fittest the data. So, when these algorithms' input is unscaled features, large scale data has more influence on the weight. It's not the reflection of data itself. The main advantage of scaling is to avoid attributes in greater numeric ranges dominating those in smaller numeric ranges

We have used MinMax Scaler to scale all the values between 0 and 1.

Modelling:

PREDICTIVE MODELLING:
(a) TRAIN-TEST SPLIT – We split the training and testing datasets, on a 75/25 ratio of training to testing datasets. After the final model has been selected the cross validation has been performed in order to get a more accurate value of the error function.

(b) ERROR FUNCTION – Sci-kitlearn's mean_squared_error() function was used and then the square root of mean squared error was taken, as per the instructions described in the PDF.

$$\sum_{t=1}^{N} (\mathrm{rt} - \mathrm{yt})^2$$

Regression Models

As the problem involves predicting the wages, we are using regression models and not the classification models.

Models Used:

1. Linear Regression and Polynomial Regression
   The basic linear regression and polynomial regression were our first choice, going by the traditional approaches.

   Rmse Score varied from 37000 to 45000.

2. Lasso Regression
   Lasso Regression from the scikit learn model was used as the 2nd Model.

   Rmse score ranged from 35000 to 43000

3. Decision Tree Regressor
   Next, we decided to try the decision tree regressor as we can tune no of splits in order to get lesser rmse. The following image describes the rmse for different leaf nodes (splitting value)

Error in Decision Tree Regressor:
Min sample leaves: 10
rmse: 42323.176654867304
Min sample leaves: 20
rmse: 37027.77805019394
Min sample leaves: 30
rmse: 35776.21661475761
Min sample leaves: 50
rmse: 34722.34808316169
Min sample leaves: 70
rmse: 39229.402442094644
Min sample leaves: 100
rmse: 37667.85308780998

4. Random Forest Regressor
   Since decision tree gave a better result, we though why not tune the no of trees as well. So, the obvious choice after Decision Tree was Random Forest Regressor and unsurprisingly, the rmse value where between 33000 to 43000.
   We experimented with different no. of splitting leaf values and also with minimum number of trees.

5. Elastic Net Regressor
   Elastic Net Regressor Model from sklearn is a great combination of Ridge and Lasso models, so we decided why not give it a try. Alas, the results weren't better than the Random Forest Regressor
   The rmse values ranged from 36000 to 44000

6. Ridge Regressor
   The Ridgre Regressor is similar to the Lasso Regressor and comes along with sklearn models.
   The rmse values ranged from 36000 to 44000

Final Model:

The final model selected is Random Forest Regressor after comparing all the rmse values of different models.

Once the model was finalized cross validation was performed to get the final root mean square error value.

10-Fold validation was performed on the dataset and the
resulting root mean square error was: 40387.39446

References

[1] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html
[2] https://scikit-learn.org/stable/modules/preprocessing.html
[3] https://pythonprogramming.net/pandas-statistics-correlation-tables-how-to/
[4] https://seaborn.pydata.org/examples/many_pairwise_correlations.html
[5] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
[6] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html
[7] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNet.html
[8] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html
[9] https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html
[10] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
[11] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html
[12] https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/