Predicting Median General Practitioner (GP) Earnings

Dev Makwana, Anirudh Chintaluri

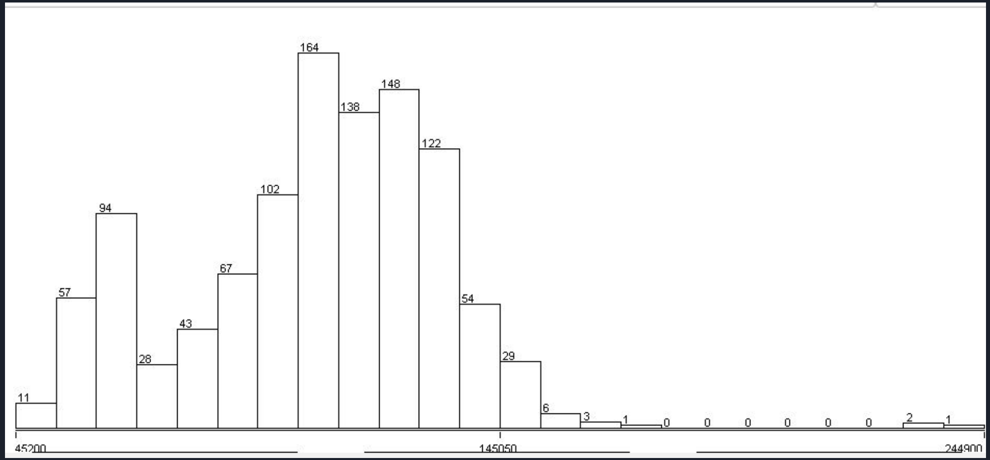
Project Statement

The purpose of this project is to create a model to predict the median income (before tax) of general practitioners (GPs) in the United Kingdom.

This can be used to better understand the circumstances for GPs in specific demographics, and can be used to make better healthcare policy-making decisions from governments and healthcare firms, as well as using it as a metric for economic forecasting, the job market, and general quality of life.

Description of the Dataset

- Single dataset from the National Health Service England Website
- 55 attributes, one class attribute, and 1406 instances (before preprocessing).



Description of the Dataset

1. GP_Type
2. Contract_Type
3. Country
4. Practice_Type
5. Gender
6. Age
7. Rurality
8. Region
9. Practice_Registered_Patients
10. Weekly Working Hours
11. Range_of_Gross_Earnings_from_Self_Employment_£
12. Range_of_Total_Expenses_from_Self_Employment_£
13. Range_of_Income_Before_Tax_from_Self_Employment_£
14. Range_of_Total_Income_Before_Tax_£
15. Sample Count
16. Estimated Population
17. Average SE Gross Earnings
18. Average SE Expenses
19. Average SE Income Before Tax
20. Average Emp Gross Earnings
21. Average Emp Expenses
22. Average Emp Income Before Tax
23. Average Tot Gross Earnings
24. Average Tot Expenses
25. Average Tot Income Before Tax
26. EER
27. Income Before Tax Standard Error
28. Median Income Before Tax
29. Average Total Expenses
30. Average Office and General Business
31. Average Premises
32. Average Employee
33. Average Car and Travel
34. Average Interest
35. Average Other
36. Average Net Capital Allowances
37. %Zero Office and General Business
38. %Zero Premises
39. %Zero Employee
40. %Zero Car and Travel
41. %Zero Interest
42. %Zero Other
43. %Zero Net Capital Allowances
44. Count_of_GPs
45. Percentage_of_GPs_%
46. Cumulative_Percentage_of_GPs_%
47. GE Median
48. GE Q1
49. GE Q3
50. GE D1
51. GE D9
52. TE Median
53. IBT Q1
54. IBT Q3
55. IBT D1
56. IBT D9

Pre-Processing

The following were the steps we took for pre-processing the data:

1. Remove any instances missing the class value

Selected attribute
Name: Median Income Before Tax
Missing: 336 (24%) Distinct: 521
Type: Numeric
Unique: 239 (17%)

2. Remove unnecessary attributes

Selected attribute
Name: Weekly Working Hours
Missing: 0 (0%) Distinct: 1
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	All	1406	1406

3. Removing attributes with too many missing values

Selected attribute
Name: GE Q1
Missing: 1386 (99%) Distinct: 16
Type: Numeric
Unique: 12 (1%)

Statistic	Value
Minimum	49600
Maximum	340600
Mean	150115
StdDev	107865.917

Pre-Processing

3. Removing attributes too similar to class

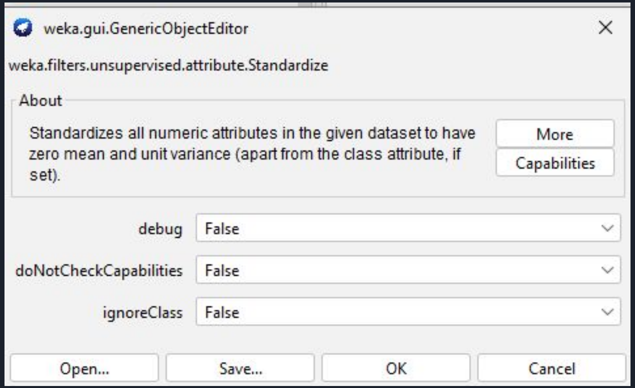
- Average Tot Gross Earnings
- Average Tot Income Before Tax
- Income Before Tax Standard Error

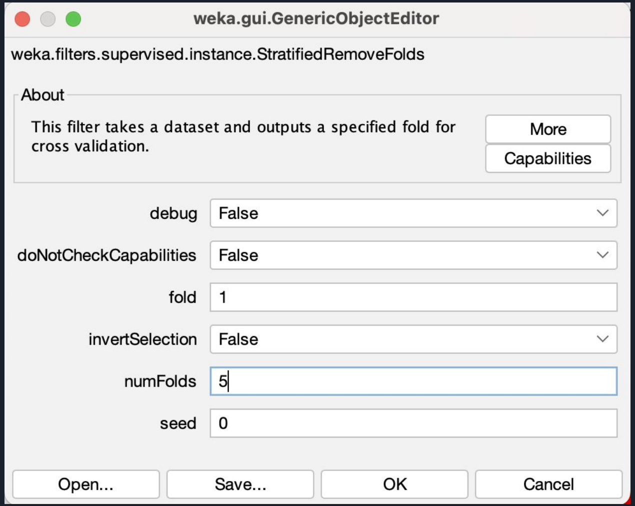
4. Normalize the data

- Using Standardize to normalize by **z-score**

5. Split the dataset into training and testing datasets with Stratified Random Sampling

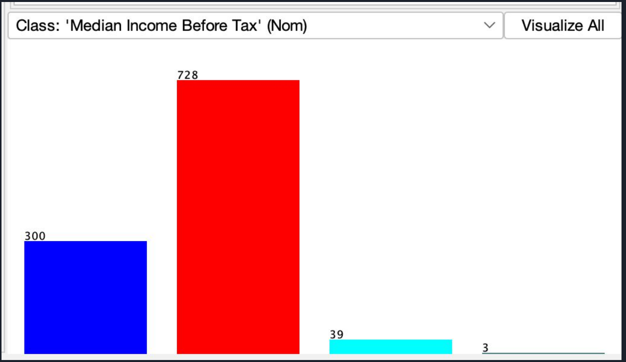
- 80% for training, 20% for testing
- invertSelection = True for training





Discretization of Class

Converted the class to a nominal data type by discretizing it into four bins of equal width.



Selected attribute

Name: 'Median Income Before Tax' Type: Nominal

Missing: 0 (0%) Distinct: 4 Unique: 0 (0%)

No.	Label	Count	Weight
1	'(-inf-95125]'	300	300
2	'(95125-145050]'	728	728
3	'(145050-194975]'	39	39
4	'(194975-inf)'	3	3

Attribute Selection Algorithms: Correlation

Finds the Pearson correlation coefficient for each attribute

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

Attributes with a correlation coefficient below 0.05 were removed. (Removed 7 attributes)

```
Ranked attributes:
 0.5473    9 'Average Tot Expenses'
 0.4567    1 GP_Type
 0.1494    4 Practice_Type
 0.1069    5 Gender
 0.0971   17 'Average Other'
 0.0836   11 'Average Total Expenses'
 0.0813    6 Age
 0.0785   10 EER
 0.069    19 '%Zero Employee'
 0.0665   14 'Average Employee'
 0.0656   12 'Average Office and General Business'
 0.0655    7 'Sample Count'
 0.0654   16 'Average Interest'
 0.06     18 Average Net Capital Allowances'
 0.0539   20 '%Zero Car and Travel'
 0.0531    3 Country
 0.0386   13 'Average Premises'
 0.0347   21 '%Zero Interest'
 0.0346    8 'Estimated Population'
 0.0169   22 %Zero Net Capital Allowances'
 0.0139   15 'Average Car and Travel'
 0.0106    2 Contract_Type
```

Attribute Selection Algorithm: OneR

Creates a rule to predict the class using a single attribute. Finds the rule with the lowest error rate using the pseudocode below.

For each attribute
   For each unique value of the attribute
      count the frequency of each class value
      find the most frequent class value
      make rule where the most frequent class value is assigned to this value of the attribute
   Calculate the error rate of each rule for this attribute
Choose the rule with the lowest error rate

Attributes with a score of less than 68.0 were removed. (Removed 6 attributes)

```
Ranked attributes:
86.63551    9 'Average Tot Expenses'
86.26168    1 GP_Type
85.98131   10 EER
78.03738   13 'Average Premises'
77.85047   17 'Average Other'
77.75701   11 'Average Total Expenses'
77.47664   12 'Average Office and General Business'
77.38318   15 'Average Car and Travel'
77.38318   18 Average Net Capital Allowances'
77.00935   16 'Average Interest'
76.82243   14 'Average Employee'
68.03738   20 '%Zero Car and Travel'
68.03738    4 Practice_Type
68.03738    2 Contract_Type
68.03738    3 Country
68.03738    5 Gender
68.03738    6 Age
67.94393   22 %Zero Net Capital Allowances'
67.94393   21 '%Zero Interest'
67.85047   19 '%Zero Employee'
67.19626    7 'Sample Count'
66.63551    8 'Estimated Population'
```

Attribute Selection Algorithm: Info Gain

Uses the following formulas to calculate Gain(A) for each attribute A

$$\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)$$

$$\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

where m is the number of classes and p_i is the probability that a tuple in dataset D belongs to C_i, estimated by taking the ratio of the number of tuples in D where the class is C_i to the total number of tuples in D. Attributes with an information gain value below 0.05 were removed.

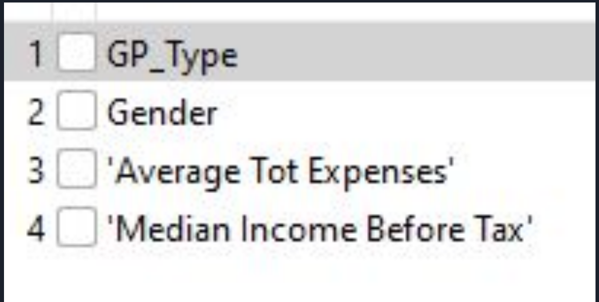
Ranked attributes:

0.58185	9	'Average Tot Expenses'
0.5561	10	EER
0.48809	1	GP_Type
0.47333	14	'Average Employee'
0.43973	16	'Average Interest'
0.42296	11	'Average Total Expenses'
0.41858	13	'Average Premises'
0.40513	18	Average Net Capital Allowances'
0.40381	15	'Average Car and Travel'
0.38211	12	'Average Office and General Business'
0.37884	17	'Average Other'
0.26645	21	'%Zero Interest'
0.20107	20	'%Zero Car and Travel'
0.17778	22	%Zero Net Capital Allowances'
0.11987	19	'%Zero Employee'
0.096	4	Practice_Type
0.03429	3	Country
0.03125	5	Gender
0.01593	7	'Sample Count'
0.0132	6	Age
0.00278	2	Contract_Type
0	8	'Estimated Population'

Attribute Selection Algorithm: Cfs Subset

Evaluates the worth of a different subsets of attributes, using the GreedyStepwise to find the best subset.

Below is the best subset found



Attribute Selection Algorithm: Hand-Picked

1	GP_Type
2	Contract_Type
3	Country
4	Practice_Type
5	Gender
6	Age
7	'Sample Count'
8	'Estimated Population'
9	'Average Tot Expenses'
10	EER
11	'Average Total Expenses'
12	'Average Office and General Business'
13	'Average Premises'
14	'Average Employee'
15	'Average Car and Travel'
16	'Average Interest'
17	'Average Other'
18	Average Net Capital Allowances'
19	'%Zero Employee'
20	'%Zero Car and Travel'
21	'%Zero Interest'
22	%Zero Net Capital Allowances'
23	'Median Income Before Tax'

Classifier Model: Naive Bayes

Calculates probabilities using **Bayes' Theorem**

- Assumes the features we use to predict the target are **independent**

$$P(C_i \mid X) = \frac{P(X \mid C_i)P(C_i)}{P(X)}$$

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \ldots \times P(x_n \mid C_i)$$

	Not Spam	Spam
Dear	8	3
Visit	2	6
Invitation	5	2
Link	2	7
Friend	6	1
Hello	5	4
Discount	0	8
Money	1	7
Click	2	9
Dinner	3	0
Total Words	34	47

The class with the **highest probability** becomes the model's prediction

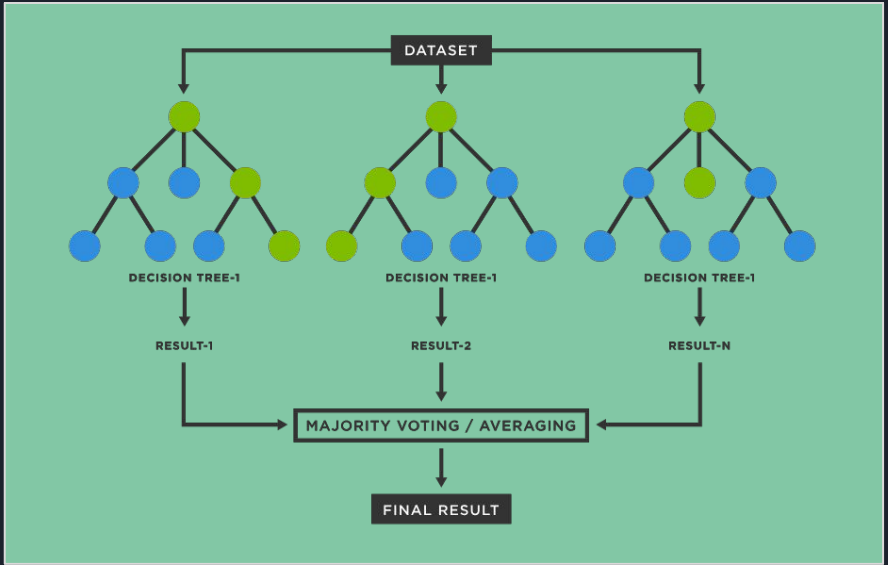
$$P(Hello\ Friend \mid Not\ Spam) = P(Hello \mid Not\ Spam) * P(Friend \mid Not\ Spam)$$

$$P(Not\ Spam \mid Hello\ Friend) = P(Hello \mid Not\ Spam) * P(Friend \mid Not\ Spam) * P(Not\ Spam)$$

$$P(Not\ Spam \mid Hello\ Friend) = \frac{5}{34} * \frac{6}{34} * \frac{15}{25} = 0.0155$$

Classifier Model: Random Forest

- **Voting**-based model
- Creates multiple decision trees to determine class prediction
- Class with **highest number of trees** "votes" becomes the model's prediction
- All of the trees are **equally weighted** for this process.



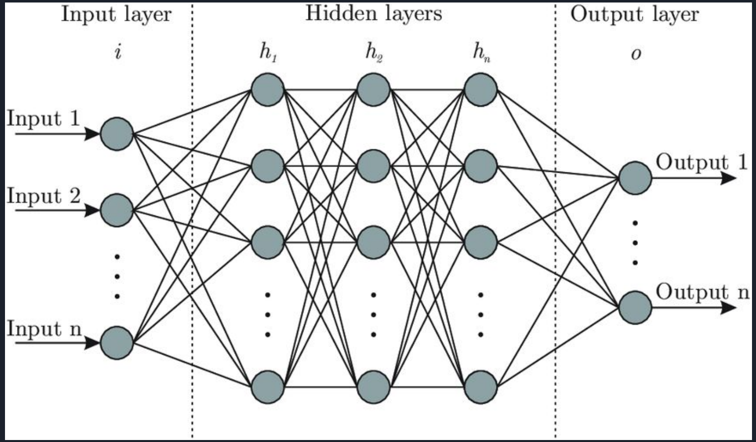
Classifier Model: OneR

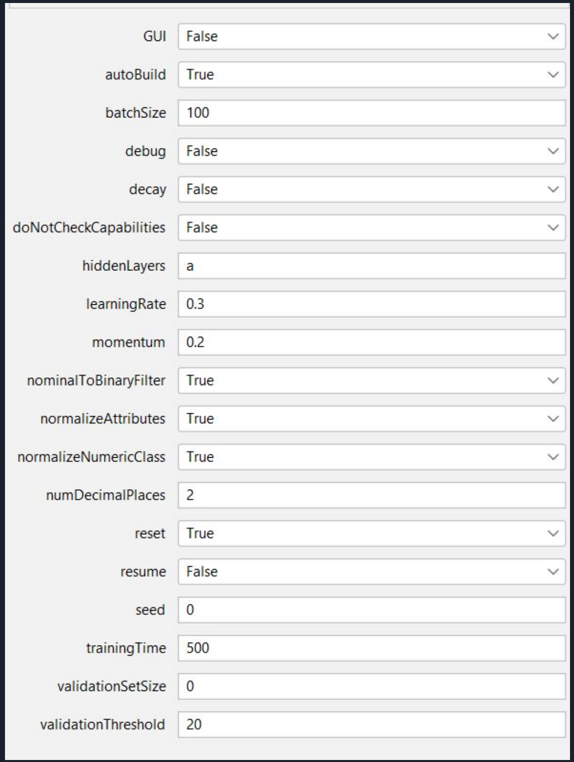
This works the **same** as the OneR attribute selection algorithm.

For each attribute
For each unique value of the attribute
count the frequency of each class value
find the most frequent class value
make rule where the most frequent class value is assigned to this value of the attribute
Calculate the error rate of each rule for this attribute
Choose the rule with the lowest error rate

Classifier Model: Multilayer Perceptron

- Simplistic **Neural Network**
- Trained through **gradient descent, backpropagation**

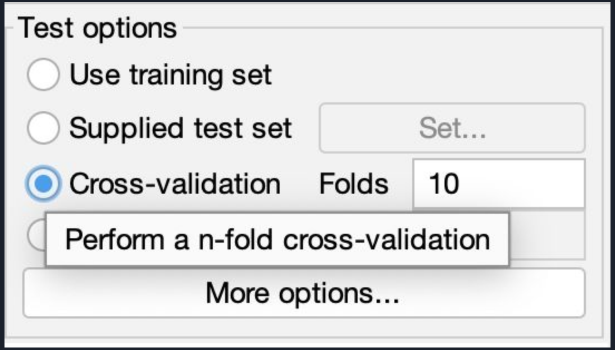




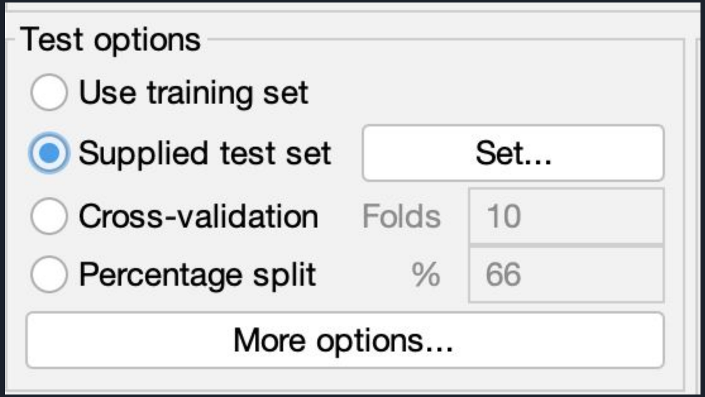
GUI	False
autoBuild	True
batchSize	100
debug	False
decay	False
doNotCheckCapabilities	False
hiddenLayers	a
learningRate	0.3
momentum	0.2
nominalToBinaryFilter	True
normalizeAttributes	True
normalizeNumericClass	True
numDecimalPlaces	2
reset	True
resume	False
seed	0
trainingTime	500
validationSetSize	0
validationThreshold	20

Measuring Performance

- Training: 10-fold **cross-validation**
  - **No need for validation set**



- Testing: Supplied Test Set



Results - Accuracy

Training, cross-validation

	Naive Bayes	Random Forest	OneR	MLP
Correlation	72.5467	95.0935	86.0981	92.5234
OneR	70.9112	93.5748	86.0981	89.6028
Info Gain	70.4439	95.0935	86.0981	94.2757
CfsSubset	87.9673	88.7850	86.4486	88.4346
Custom	70.2103	94.5093	86.0981	90.0701

Testing, testing dataset

	Naive Bayes	Random Forest	OneR	MLP
Correlation	70.8738	97.5728	88.3495	95.1456
OneR	66.9903	97.0874	88.3495	97.0874
Info Gain	67.4757	96.6019	88.3495	96.6019
CfsSubset	91.7476	94.1748	88.3495	92.2330
Custom	68.4466	95.1456	88.3495	96.6019

Results - TPR, FPR

True Positive Rate (TPR)

	Naive Bayes	Random Forest	OneR	MLP
Correlation	0.725	0.951	0.861	0.925
OneR	0.709	0.936	0.861	0.896
Info Gain	0.704	0.951	0.861	0.943
CfsSubset	0.880	0.888	0.864	0.884
Custom	0.702	0.945	0.861	0.901

False Positive Rate (FPR)

	Naive Bayes	Random Forest	OneR	MLP
Correlation	0.146	0.077	0.232	0.115
OneR	0.139	0.101	0.232	0.175
Info Gain	0.146	0.077	0.232	0.089
CfsSubset	0.211	0.118	0.225	0.214
Custom	0.151	0.085	0.232	0.162

Results - ROC Area

	Naive Bayes	Random Forest	OneR	MLP
Correlation	0.882	0.987	0.814	0.972
OneR	0.895	0.986	0.814	0.952
Info Gain	0.891	0.984	0.814	0.965
CfsSubset	0.915	0.939	0.820	0.920
Custom	0.890	0.987	0.814	0.947

Results

```
Time taken to build model: 0.1 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         210               98.1308 %
Incorrectly Classified Instances         4                1.8692 %
Kappa statistic                          0.9585
Mean absolute error                      0.0263
Root mean squared error                  0.0911
Relative absolute error                 11.4989 %
Root relative squared error             27.0137 %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.950    0.006    0.983      0.950    0.966      0.953    0.999     0.998     '(-inf-95
                 0.993    0.044    0.980      0.993    0.986      0.957    0.999     0.999     '(95125-1
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     '(145050-
                 ?        0.000    ?          ?        ?          ?        ?         ?         '(194975-
Weighted Avg.    0.981    0.032    0.981      0.981    0.981      0.957    0.999     0.999

=== Confusion Matrix ===

   a   b   c   d   <-- classified as
  57   3   0   0 |   a = '(-inf-95125]'
   1 145   0   0 |   b = '(95125-145050]'
   0   0   8   0 |   c = '(145050-194975]'
   0   0   0   0 |   d = '(194975-inf)'
```

Analysis

- **Pearson Correlation** approach most effective attribute selection algorithm
- **Random Forest** most effective model
- Scores ranged from 66.9903% to 97.5728%
- Chose **Correlation-RandomForest** as our model with the highest accuracy, the lowest error rates of all models, highest TP, lowest FP, and near perfect ROC area.
- Testing accuracy scores **inflated**

```
Time taken to build model: 0.07 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         201               97.5728 %
Incorrectly Classified Instances         5                2.4272 %
Kappa statistic                          0.9413
Mean absolute error                      0.0377
Root mean squared error                  0.1145
Relative absolute error                 17.1982 %
Root relative squared error             35.5224 %
Total Number of Instances              206

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.950    0.000    1.000      0.950    0.974      0.965    0.998     0.996     '(-inf-95125]'
                 0.986    0.050    0.980      0.986    0.983      0.941    0.995     0.998     '(95125-145050]'
                 ?        0.010    0.000      ?        ?          ?        ?         ?         '(145050-194975]'
                 ?        0.000    ?          ?        ?          ?        ?         ?         '(194975-inf)'
Weighted Avg.    0.976    0.035    0.986      0.976    0.980      0.948    0.996     0.997

=== Confusion Matrix ===

  a   b   c   d   <-- classified as
 57   3   0   0 |   a = '(-inf-95125]'
  0 144   2   0 |   b = '(95125-145050]'
  0   0   0   0 |   c = '(145050-194975]'
  0   0   0   0 |   d = '(194975-inf)'
```

Conclusion

- We were able to successfully train machine learning models to predict salary of GPs based on demographics
- All models achieved **at least 65% accuracy**
- Highest performing model: **Correlation-RandomForest** - 97.5728% accuracy
- Limitations:
  - **Last class bin not represented** well in the test set (3 total instances)
- Future Directions:
  - Collect more data such that those of the **highest income batch are represented** well in the dataset
    - SMOTE as an alternative
  - Enables policy-makers to make better decisions for development of society