DATA MINING LAB MANUAL

Subtasks :

1. List all the categorical (or nominal) attributes and the real-valued attributes seperately.

Attributes:-

1. checking_status
2. duration
3. credit history
4. purpose
5. credit amount
6. savings_status
7. employment duration
8. installment rate
9. personal status
10. debitors
11. residence_since
12. property
14. installment plans
15. housing
16. existing credits
17. job
18. num_dependents
19. telephone
20. foreign worker

Categorical or Nomianal attributes:-

1. checking_status
2. credit history
3. purpose
4. savings_status
5. employment
6. personal status
7. debtors
8. property
9. installment plans
10. housing
11. job
12. telephone
13. foreign worker

Real valued attributes:-

1. duration
2. credit amount
3. credit amount
4. residence

5. age
6. existing credits
7. num_dependents

2. What attributes do you think might be crucial in making the credit assessement ? Come up with some simple rules in plain English using your selected attributes.

According to me the following attributes may be crucial in making the credit risk assessment.

1. Credit_history
2. Employment
3. Property_magnitude
4. job
5. duration
6. crdit_amount
7. installment
8. existing credit

Basing on the above attributes, we can make a decision whether to give credit or not.

3. One type of model that you can create is a Decision Tree - train a Decision Tree using the complete dataset as the training data. Report the model obtained after training.

```
J48 pruned tree
------------------

checking_status = <0
|   foreign_worker = yes
|   |   duration <= 11
|   |   |   existing_credits <= 1
|   |   |   |   property_magnitude = real estate: good (8.0/1.0)
|   |   |   |   property_magnitude = life insurance
|   |   |   |   |   own_telephone = none: bad (2.0)
|   |   |   |   |   own_telephone = yes: good (4.0)
|   |   |   |   property_magnitude = car: good (2.0/1.0)
|   |   |   |   property_magnitude = no known property: bad (3.0)
|   |   |   existing_credits > 1: good (14.0)
|   |   duration > 11
|   |   |   job = unemp/unskilled non res: bad (5.0/1.0)
|   |   |   job = unskilled resident
|   |   |   |   purpose = new car
|   |   |   |   |   own_telephone = none: bad (10.0/2.0)
|   |   |   |   |   own_telephone = yes: good (2.0)
|   |   |   |   purpose = used car: bad (1.0)
|   |   |   |   purpose = furniture/equipment
|   |   |   |   |   employment = unemployed: good (0.0)
|   |   |   |   |   employment = <1: bad (3.0)
|   |   |   |   |   employment = 1<=X<4: good (4.0)
|   |   |   |   |   employment = 4<=X<7: good (1.0)
|   |   |   |   |   employment = >=7: good (2.0)
|   |   |   |   purpose = radio/tv
|   |   |   |   |   existing_credits <= 1: bad (10.0/3.0)
|   |   |   |   |   existing_credits > 1: good (2.0)
|   |   |   |   purpose = domestic appliance: bad (1.0)
|   |   |   |   purpose = repairs: bad (1.0)
|   |   |   |   purpose = education: bad (1.0)
|   |   |   |   purpose = vacation: bad (0.0)
|   |   |   |   purpose = retraining: good (1.0)
|   |   |   |   purpose = business: good (3.0)
|   |   |   |   purpose = other: good (1.0)
|   |   |   job = skilled
|   |   |   |   other_parties = none
|   |   |   |   |   duration <= 30
|   |   |   |   |   |   savings_status = <100
|   |   |   |   |   |   |   credit_history = no credits/all paid: bad (8.0/1.0)
|   |   |   |   |   |   |   credit_history = all paid: bad (6.0)
|   |   |   |   |   |   |   credit_history = existing paid
|   |   |   |   |   |   |   |   own_telephone = none
|   |   |   |   |   |   |   |   |   existing_credits <= 1
|   |   |   |   |   |   |   |   |   |   property_magnitude = real estate
|   |   |   |   |   |   |   |   |   |   |   age <= 26: bad (5.0)
|   |   |   |   |   |   |   |   |   |   |   age > 26: good (2.0)
|   |   |   |   |   |   |   |   |   |   property_magnitude = life insurance: bad (7.0/2.0)
|   |   |   |   |   |   |   |   |   |   property_magnitude = car
|   |   |   |   |   |   |   |   |   |   |   credit_amount <= 1386: bad (3.0)
|   |   |   |   |   |   |   |   |   |   |   credit_amount > 1386: good (11.0/1.0)
|   |   |   |   |   |   |   |   |   |   property_magnitude = no known property: good (2.0)
|   |   |   |   |   |   |   |   |   existing_credits > 1: bad (3.0)
|   |   |   |   |   |   |   |   own_telephone = yes: bad (5.0)
|   |   |   |   |   |   |   credit_history = delayed previously: bad (4.0)
|   |   |   |   |   |   |   credit_history = critical/other existing credit: good (14.0/4.0)
|   |   |   |   |   |   savings_status = 100<=X<500
|   |   |   |   |   |   |   credit_history = no credits/all paid: good (0.0)
|   |   |   |   |   |   |   credit_history = all paid: good (1.0)
|   |   |   |   |   |   |   credit_history = existing paid: bad (3.0)
|   |   |   |   |   |   |   credit_history = delayed previously: good (0.0)
|   |   |   |   |   |   |   credit_history = critical/other existing credit: good (2.0)
|   |   |   |   |   |   savings_status = 500<=X<1000: good (4.0/1.0)
|   |   |   |   |   |   savings_status = >=1000: good (4.0)
|   |   |   |   |   |   savings_status = no known savings
|   |   |   |   |   |   |   existing_credits <= 1
|   |   |   |   |   |   |   |   own_telephone = none: bad (9.0/1.0)
|   |   |   |   |   |   |   |   own_telephone = yes: good (4.0/1.0)
|   |   |   |   |   |   |   existing_credits > 1: good (2.0)
|   |   |   |   |   duration > 30: bad (30.0/3.0)
|   |   |   |   other_parties = co applicant: bad (7.0/1.0)
|   |   |   |   other_parties = guarantor: good (12.0/3.0)
|   |   |   job = high qualif/self emp/mgmt: good (30.0/8.0)
|   foreign_worker = no: good (15.0/2.0)
checking_status = 0<=X<200
|   credit_amount <= 9857
|   |   savings_status = <100
|   |   |   other_parties = none
|   |   |   |   duration <= 42
|   |   |   |   |   personal_status = male div/sep: bad (8.0/2.0)
|   |   |   |   |   personal_status = female div/dep/mar
|   |   |   |   |   |   purpose = new car: bad (5.0/1.0)
|   |   |   |   |   |   purpose = used car: bad (1.0)
|   |   |   |   |   |   purpose = furniture/equipment
|   |   |   |   |   |   |   duration <= 10: bad (3.0)
|   |   |   |   |   |   |   duration > 10
|   |   |   |   |   |   |   |   duration <= 21: good (6.0/1.0)
|   |   |   |   |   |   |   |   duration > 21: bad (2.0)
|   |   |   |   |   |   purpose = radio/tv: good (8.0/2.0)
|   |   |   |   |   |   purpose = domestic appliance: good (0.0)
|   |   |   |   |   |   purpose = repairs: good (1.0)
|   |   |   |   |   |   purpose = education: good (4.0/2.0)
|   |   |   |   |   |   purpose = vacation: good (0.0)
|   |   |   |   |   |   purpose = retraining: good (0.0)
|   |   |   |   |   |   purpose = business
|   |   |   |   |   |   |   residence_since <= 2: good (3.0)
|   |   |   |   |   |   |   residence_since > 2: bad (2.0)
|   |   |   |   |   |   purpose = other: good (0.0)
|   |   |   |   |   personal_status = male single: good (52.0/15.0)
|   |   |   |   |   personal_status = male mar/wid
|   |   |   |   |   |   duration <= 10: good (6.0)
|   |   |   |   |   |   duration > 10: bad (10.0/3.0)
|   |   |   |   |   personal_status = female single: good (0.0)
|   |   |   |   duration > 42: bad (7.0)
|   |   |   other_parties = co applicant: good (2.0)
|   |   |   other_parties = guarantor
|   |   |   |   purpose = new car: bad (2.0)
|   |   |   |   purpose = used car: good (0.0)
|   |   |   |   purpose = furniture/equipment: good (0.0)
|   |   |   |   purpose = radio/tv: good (18.0/1.0)
|   |   |   |   purpose = domestic appliance: good (0.0)
|   |   |   |   purpose = repairs: good (0.0)
|   |   |   |   purpose = education: good (0.0)
|   |   |   |   purpose = vacation: good (0.0)
|   |   |   |   purpose = retraining: good (0.0)
|   |   |   |   purpose = business: good (0.0)
|   |   |   |   purpose = other: good (0.0)
|   |   savings_status = 100<=X<500
|   |   |   purpose = new car: bad (15.0/5.0)
|   |   |   purpose = used car: good (3.0)
|   |   |   purpose = furniture/equipment: bad (4.0/1.0)
|   |   |   purpose = radio/tv: bad (8.0/2.0)
|   |   |   purpose = domestic appliance: good (0.0)
|   |   |   purpose = repairs: good (2.0)
|   |   |   purpose = education: good (0.0)
|   |   |   purpose = vacation: good (0.0)
|   |   |   purpose = retraining: good (0.0)
|   |   |   purpose = business
|   |   |   |   housing = rent
|   |   |   |   |   existing_credits <= 1: good (2.0)
|   |   |   |   |   existing_credits > 1: bad (2.0)
|   |   |   |   housing = own: good (6.0)
|   |   |   |   housing = for free: bad (1.0)
|   |   |   purpose = other: good (1.0)
|   |   savings_status = 500<=X<1000: good (11.0/3.0)
|   |   savings_status = >=1000: good (13.0/3.0)
|   |   savings_status = no known savings: good (41.0/5.0)
|   credit_amount > 9857: bad (20.0/3.0)
checking_status = >=200: good (63.0/14.0)
checking_status = no checking: good (394.0/46.0)

Number of Leaves  :     103

Size of the tree :      140


Time taken to build model: 0.03 seconds

=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances         855               85.5    %
Incorrectly Classified Instances       145               14.5    %
Kappa statistic                          0.6251
Mean absolute error                      0.2312
Root mean squared error                  0.34
Relative absolute error                 55.0377 %
Root relative squared error             74.2015 %
Total Number of Instances             1000
```

4. Suppose you use your above model trained on the complete dataset, and classify credit good/bad for each of the examples in the dataset. What % of examples can you classify correctly? (This is also called testing on the training set) Why do you think you cannot get 100 % training accuracy?

In the above model we trained complete dataset and we classified credit good/bad for each of the examples in the dataset.

For example:

```
IF
    purpose=vacation THEN
            credit=bad
ELSE
    purpose=business THEN
            Credit=good
```

In this way we classified each of the examples in the dataset.

We classified 85.5% of examples correctly and the remaining 14.5% of examples are incorrectly classified. We can't get 100% training accuracy because out of the 20 attributes, we have some unnecessary attributes which are also been analyzed and trained.

Due to this the accuracy is affected and hence we can't get 100% training accuracy.

5. Is testing on the training set as you did above a good idea? Why or Why not?

According to the rules, for the maximum accuracy, we have to take 2/3 of the dataset as training set and the remaining 1/3 as test set. But here in the above model we have taken complete dataset as training set which results only 85.5% accuracy.

This is done for the analyzing and training of the unnecessary attributes which does not make a crucial role in credit risk assessment. And by this complexity is increasing and finally it leads to the minimum accuracy.

If some part of the dataset is used as a training set and the remaining as test set then it leads to the accurate results and the time for computation will be less.

This is why, we prefer not to take complete dataset as training set.

6. One approach for solving the problem encountered in the previous question is using cross-validation? Describe what cross-validation is briefly. Train a Decision Tree again using cross-validation and report your results. Does your accuracy increase/decrease? Why? (10 marks)

Cross validation:-

In k-fold cross-validation, the initial data are randomly portioned into 'k' mutually exclusive subsets or folds D1, D_2, D_3, ..., D_k. Each of approximately equal size. Training and testing is performed 'k' times. In iteration I, partition D_i is reserved as the test set and the remaining partitions are collectively used to train the model. That is in the first iteration subsets D_2, D_3, ..., D_k collectively serve as the training set in order to obtain as first model. Which is tested on D_i. The second trained on the subsets D_1, D_3, ..., D_k and test on the D_2 and so on….

```
J48 pruned tree :-
------------------

checking_status = <0
|  foreign_worker = yes
|  |  duration <= 11
|  |  |  existing_credits <= 1
|  |  |  |  property_magnitude = real estate: good (8.0/1.0)
|  |  |  |  property_magnitude = life insurance
|  |  |  |  |  own_telephone = none: bad (2.0)
|  |  |  |  |  own_telephone = yes: good (4.0)
|  |  |  |  property_magnitude = car: good (2.0/1.0)
|  |  |  |  property_magnitude = no known property: bad (3.0)
|  |  |  existing_credits > 1: good (14.0)
|  |  duration > 11
|  |  |  job = unemp/unskilled non res: bad (5.0/1.0)
|  |  |  job = unskilled resident
|  |  |  |  purpose = new car
|  |  |  |  |  own_telephone = none: bad (10.0/2.0)
|  |  |  |  |  own_telephone = yes: good (2.0)
|  |  |  |  purpose = used car: bad (1.0)
|  |  |  |  purpose = furniture/equipment
|  |  |  |  |  employment = unemployed: good (0.0)
|  |  |  |  |  employment = <1: bad (3.0)
|  |  |  |  |  employment = 1<=X<4: good (4.0)
|  |  |  |  |  employment = 4<=X<7: good (1.0)
|  |  |  |  |  employment = >=7: good (2.0)
|  |  |  |  purpose = radio/tv
|  |  |  |  |  existing_credits <= 1: bad (10.0/3.0)
|  |  |  |  |  existing_credits > 1: good (2.0)
|  |  |  |  purpose = domestic appliance: bad (1.0)
|  |  |  |  purpose = repairs: bad (1.0)
|  |  |  |  purpose = education: bad (1.0)
|   |   |   |   purpose = vacation: bad (0.0)
|   |   |   |   purpose = retraining: good (1.0)
|   |   |   |   purpose = business: good (3.0)
|   |   |   |   purpose = other: good (1.0)
|   |   |   job = skilled
|   |   |   |   other_parties = none
|   |   |   |   |   duration <= 30
|   |   |   |   |   |   savings_status = <100
|   |   |   |   |   |   |   credit_history = no credits/all paid: bad (8.0/1.0)
|   |   |   |   |   |   |   credit_history = all paid: bad (6.0)
|   |   |   |   |   |   |   credit_history = existing paid
|   |   |   |   |   |   |   |   own_telephone = none
|   |   |   |   |   |   |   |   |   existing_credits <= 1
|   |   |   |   |   |   |   |   |   |   property_magnitude = real estate
|   |   |   |   |   |   |   |   |   |   |   age <= 26: bad (5.0)
|   |   |   |   |   |   |   |   |   |   |   age > 26: good (2.0)
|   |   |   |   |   |   |   |   |   |   property_magnitude = life insurance: bad (7.0/2.0)
|   |   |   |   |   |   |   |   |   |   property_magnitude = car
|   |   |   |   |   |   |   |   |   |   |   credit_amount <= 1386: bad (3.0)
|   |   |   |   |   |   |   |   |   |   |   credit_amount > 1386: good (11.0/1.0)
|   |   |   |   |   |   |   |   |   |   property_magnitude = no known property: good (2.0)
|   |   |   |   |   |   |   |   |   existing_credits > 1: bad (3.0)
|   |   |   |   |   |   |   |   own_telephone = yes: bad (5.0)
|   |   |   |   |   |   |   credit_history = delayed previously: bad (4.0)
|   |   |   |   |   |   |   credit_history = critical/other existing credit: good (14.0/4.0)
|   |   |   |   |   |   savings_status = 100<=X<500
|   |   |   |   |   |   |   credit_history = no credits/all paid: good (0.0)
|   |   |   |   |   |   |   credit_history = all paid: good (1.0)
|   |   |   |   |   |   |   credit_history = existing paid: bad (3.0)
|   |   |   |   |   |   |   credit_history = delayed previously: good (0.0)
|   |   |   |   |   |   |   credit_history = critical/other existing credit: good (2.0)
|   |   |   |   |   |   savings_status = 500<=X<1000: good (4.0/1.0)
|   |   |   |   |   |   savings_status = >=1000: good (4.0)
|   |   |   |   |   |   savings_status = no known savings
|   |   |   |   |   |   |   existing_credits <= 1
|   |   |   |   |   |   |   |   own_telephone = none: bad (9.0/1.0)
|   |   |   |   |   |   |   |   own_telephone = yes: good (4.0/1.0)
|   |   |   |   |   |   |   existing_credits > 1: good (2.0)
|   |   |   |   |   duration > 30: bad (30.0/3.0)
|   |   |   |   other_parties = co applicant: bad (7.0/1.0)
|   |   |   |   other_parties = guarantor: good (12.0/3.0)
|   |   |   job = high qualif/self emp/mgmt: good (30.0/8.0)
|   foreign_worker = no: good (15.0/2.0)
checking_status = 0<=X<200
|   credit_amount <= 9857
|   |   savings_status = <100
|   |   |   other_parties = none
|   |   |   |   duration <= 42
|   |   |   |   |   personal_status = male div/sep: bad (8.0/2.0)
|   |   |   |   |   personal_status = female div/dep/mar
|   |   |   |   |   |   purpose = new car: bad (5.0/1.0)
|   |   |   |   |   |   purpose = used car: bad (1.0)
|   |   |   |   |   |   purpose = furniture/equipment
|   |   |   |   |   |   |   duration <= 10: bad (3.0)
|   |   |   |   |   |   |   duration > 10
|   |   |   |   |   |   |   |   duration <= 21: good (6.0/1.0)
|   |   |   |   |   |   |   |   duration > 21: bad (2.0)
|   |   |   |   |   |   purpose = radio/tv: good (8.0/2.0)
|   |   |   |   |   |   purpose = domestic appliance: good (0.0)
|   |   |   |   |   |   purpose = repairs: good (1.0)
|   |   |   |   |   |   purpose = education: good (4.0/2.0)
|   |   |   |   |   |   purpose = vacation: good (0.0)
|   |   |   |   |   |   purpose = retraining: good (0.0)
|   |   |   |   |   |   purpose = business
|   |   |   |   |   |   |   residence_since <= 2: good (3.0)
|   |   |   |   |   |   |   residence_since > 2: bad (2.0)
|   |   |   |   |   |   purpose = other: good (0.0)
|   |   |   |   |   personal_status = male single: good (52.0/15.0)
|   |   |   |   |   personal_status = male mar/wid
|   |   |   |   |   |   duration <= 10: good (6.0)
|   |   |   |   |   |   duration > 10: bad (10.0/3.0)
|   |   |   |   |   personal_status = female single: good (0.0)
|   |   |   |   duration > 42: bad (7.0)
|   |   |   other_parties = co applicant: good (2.0)
|   |   |   other_parties = guarantor
|   |   |   |   purpose = new car: bad (2.0)
|   |   |   |   purpose = used car: good (0.0)
|   |   |   |   purpose = furniture/equipment: good (0.0)
|   |   |   |   purpose = radio/tv: good (18.0/1.0)
|   |   |   |   purpose = domestic appliance: good (0.0)
|   |   |   |   purpose = repairs: good (0.0)
|   |   |   |   purpose = education: good (0.0)
|   |   |   |   purpose = vacation: good (0.0)
|   |   |   |   purpose = retraining: good (0.0)
|   |   |   |   purpose = business: good (0.0)
|   |   |   |   purpose = other: good (0.0)
|   |   savings_status = 100<=X<500
|   |   |   purpose = new car: bad (15.0/5.0)
|   |   |   purpose = used car: good (3.0)
|   |   |   purpose = furniture/equipment: bad (4.0/1.0)
|   |   |   purpose = radio/tv: bad (8.0/2.0)
|   |   |   purpose = domestic appliance: good (0.0)
|   |   |   purpose = repairs: good (2.0)
|   |   |   purpose = education: good (0.0)
|   |   |   purpose = vacation: good (0.0)
|   |   |   purpose = retraining: good (0.0)
|   |   |   purpose = business
|   |   |   |   housing = rent
|   |   |   |   |   existing_credits <= 1: good (2.0)
|   |   |   |   |   existing_credits > 1: bad (2.0)
|   |   |   |   housing = own: good (6.0)
|   |   |   |   housing = for free: bad (1.0)
|   |   |   purpose = other: good (1.0)
|   |   savings_status = 500<=X<1000: good (11.0/3.0)
|   |   savings_status = >=1000: good (13.0/3.0)
|   |   savings_status = no known savings: good (41.0/5.0)
|   credit_amount > 9857: bad (20.0/3.0)
checking_status = >=200: good (63.0/14.0)
checking_status = no checking: good (394.0/46.0)

Number of Leaves  :     103

Size of the tree :      140


Time taken to build model: 0.07 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         705               70.5    %
Incorrectly Classified Instances       295               29.5    %
Kappa statistic                          0.2467
Mean absolute error                      0.3467
Root mean squared error                  0.4796
Relative absolute error                 82.5233 %
Root relative squared error            104.6565 %
Total Number of Instances             1000


=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.84      0.61      0.763     0.84      0.799      0.639    good
                 0.39      0.16      0.511     0.39      0.442      0.639    bad
Weighted Avg.    0.705     0.475     0.687     0.705     0.692      0.639

=== Confusion Matrix ===

   a   b   <-- classified as
 588 112 |   a = good
 183 117 |   b = bad
```

7. Check to see if the data shows a bias against "foreign workers" (attribute 20),or "personal-status"(attribute 9). One way to do this (perhaps rather simple minded) is to remove these attributes from the dataset and see if the decision tree created in those cases is significantly different from the full dataset case which you have already done. To remove an attribute you can use the reprocess tab in Weka's GUI Explorer. Did removing these attributes have any significant effect? Discuss.

This increase in accuracy is because thus two attributes are not much important in training and analyzing by removing this, the time has been reduced to some extent and then it results in increase in the accuracy.

The decision tree which is created is very large compared to the decision tree which we have trained now. This is the main difference between these two decision trees.

8. Another question might be, do you really need to input so many attributes to get good results? Maybe only a few would do. For example, you could try just having attributes 2, 3, 5, 7, 10, 17 (and 21, the class attribute (naturally)). Try out some combinations. (You had removed two attributes in problem 7. Remember to reload the arff data file to get all the attributes initially before you start selecting the ones you want.)

```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

credit_history = no credits/all paid: bad (40.0/15.0)
credit_history = all paid
|   employment = unemployed
|   |   duration <= 36: bad (3.0)
|   |   duration > 36: good (2.0)
|   employment = <1
|   |   duration <= 26: bad (7.0/1.0)
|   |   duration > 26: good (2.0)
|   employment = 1<=X<4: good (15.0/6.0)
|   employment = 4<=X<7: bad (10.0/4.0)
|   employment = >=7
|   |   job = unemp/unskilled non res: bad (0.0)
|   |   job = unskilled resident: good (3.0)
|   |   job = skilled: bad (3.0)
|   |   job = high qualif/self emp/mgmt: bad (4.0)
credit_history = existing paid
```

```
|   credit_amount <= 8648
|   |   duration <= 40: good (476.0/130.0)
|   |   duration > 40: bad (27.0/8.0)
|   credit_amount > 8648: bad (27.0/7.0)
credit_history = delayed previously
|   employment = unemployed
|   |   credit_amount <= 2186: bad (4.0/1.0)
|   |   credit_amount > 2186: good (2.0)
|   employment = <1
|   |   duration <= 18: good (2.0)
|   |   duration > 18: bad (10.0/2.0)
|   employment = 1<=X<4: good (33.0/6.0)
|   employment = 4<=X<7
|   |   credit_amount <= 4530
|   |   |   credit_amount <= 1680: good (3.0)
|   |   |   credit_amount > 1680: bad (3.0)
|   |   credit_amount > 4530: good (11.0)
|   employment = >=7
|   |   job = unemp/unskilled non res: good (0.0)
|   |   job = unskilled resident: good (2.0/1.0)
|   |   job = skilled: good (14.0/4.0)
|   |   job = high qualif/self emp/mgmt: bad (4.0/1.0)
credit_history = critical/other existing credit: good (293.0/50.0)

Number of Leaves  :       27

Size of the tree :        40


Time taken to build model: 0.01 seconds

=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances         764               76.4    %
Incorrectly Classified Instances       236               23.6    %
Kappa statistic                          0.3386
Mean absolute error                      0.3488
Root mean squared error                  0.4176
Relative absolute error                 83.0049 %
Root relative squared error             91.1243 %
Total Number of Instances             1000
```

```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

credit_history = no credits/all paid: bad (40.0/15.0)
credit_history = all paid
|   employment = unemployed
|   |   duration <= 36: bad (3.0)
|   |   duration > 36: good (2.0)
|   employment = <1
|   |   duration <= 26: bad (7.0/1.0)
|   |   duration > 26: good (2.0)
|   employment = 1<=X<4: good (15.0/6.0)
|   employment = 4<=X<7: bad (10.0/4.0)
|   employment = >=7
|   |   job = unemp/unskilled non res: bad (0.0)
|   |   job = unskilled resident: good (3.0)
|   |   job = skilled: bad (3.0)
|   |   job = high qualif/self emp/mgmt: bad (4.0)
credit_history = existing paid
|   credit_amount <= 8648
|   |   duration <= 40: good (476.0/130.0)
|   |   duration > 40: bad (27.0/8.0)
|   credit_amount > 8648: bad (27.0/7.0)
credit_history = delayed previously
|   employment = unemployed
|   |   credit_amount <= 2186: bad (4.0/1.0)
|   |   credit_amount > 2186: good (2.0)
|   employment = <1
|   |   duration <= 18: good (2.0)
|   |   duration > 18: bad (10.0/2.0)
|   employment = 1<=X<4: good (33.0/6.0)
|   employment = 4<=X<7
|   |   credit_amount <= 4530
|   |   |   credit_amount <= 1680: good (3.0)
|   |   |   credit_amount > 1680: bad (3.0)
|   |   credit_amount > 4530: good (11.0)
|   employment = >=7
|   |   job = unemp/unskilled non res: good (0.0)
|   |   job = unskilled resident: good (2.0/1.0)
|   |   job = skilled: good (14.0/4.0)
|   |   job = high qualif/self emp/mgmt: bad (4.0/1.0)
credit_history = critical/other existing credit: good (293.0/50.0)
```

```
Number of Leaves  :     27

Size of the tree :      40


Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances       703          70.3   %
Incorrectly Classified Instances     297          29.7   %
Kappa statistic                      0.1759
Mean absolute error                  0.3862
Root mean squared error              0.4684
Relative absolute error              91.9029 %
Root relative squared error          102.2155 %
Total Number of Instances            1000
```

9. Sometimes, the cost of rejecting an applicant who actually has a good credit (case 1) might be higher than accepting an applicant who has bad credit (case 2). Instead of counting the misclassifications equally in both cases, give a higher cost to the first case (say cost 5) and lower cost to the second case. You can do this by using a cost matrix in Weka. Train your Decision Tree again and report the Decision Tree and cross-validation results. Are they significantly different from results obtained in problem 6 (using equal cost)?

In the Problem 6, we used equal cost and we trained the decision tree. But here, we consider two cases with different cost.

Let us take cost 5 in case 1 and cost 2 in case 2.

When we give such costs in both cases and after training the decision tree, we can observe that almost equal to that of the decision tree obtained in problem 6.

But we find some difference in cost factor which is in summary in the difference in cost factor.

	Case1 (cost 5)	Case2 (cost 5)
Total Cost	3820	1705
Average cost	3.82	1.705

We don't find this cost factor in problem 6. As there we use equal cost. This is the major difference between the results of problem 6 and problem 9.

The cost matrices we used here:

Case 1: $\begin{matrix} 5 & 1 \\ 1 & 5 \end{matrix}$ Case 2: $\begin{matrix} 2 & 1 \\ 1 & 2 \end{matrix}$

10. Do you think it is a good idea to prefer simple decision trees instead of having long complex decision trees? How does the complexity of a Decision Tree relate to the bias of the model?

When we consider long complex decision trees, we will have many unnecessary attributes in the tree which results in increase of the bias of the model. Because of this, the accuracy of the model can also effected.

This problem can be reduced by considering simple decision tree. The attributes will be less and it decreases the bias of the model. Due to this the result will be more accurate.

So it is a good idea to prefer simple decision trees instead of long complex trees.

11. You can make your Decision Trees simpler by pruning the nodes. One approach is to use Reduced Error Pruning - Explain this idea briefly. Try reduced error pruning for training your Decision Trees using cross-validation (you can do this in Weka) and report the Decision Tree you obtain ? Also, report your accuracy using the pruned model. Does your accuracy increase ?

Reduced-error pruning :-

The idea of using a separate pruning set for pruning—which is applicable to decision trees as well as rule sets—is called reduced-error pruning. The variant described previously prunes a rule immediately after it has been grown and is called incremental reduced-error pruning. Another possibility is to build a full, unpruned rule set first, pruning it afterwards by discarding individual tests.

However, this method is much slower. Of course, there are many different ways to assess the worth of a rule based on the pruning set. A simple measure is to consider how well the rule would do at discriminating the predicted class from other classes if it were the only rule in the theory, operating under the closed world assumption. If it gets p instances right out of the t instances that it covers, and there are P instances of this class out of a total T of instances altogether, then it gets p positive instances right. The instances that it does not cover include N - n negative ones, where n = t – p is the number of negative instances that the rule covers and N = T - P is the total number of negative instances. Thus the rule has an overall success ratio of [p +(N - n)] T , and this quantity, evaluated on the test set, has been used to evaluate the success of a rule when using reduced-error pruning.

```
J48 pruned tree
------------------

checking_status = <0
|   foreign_worker = yes
|   |   credit_history = no credits/all paid: bad (11.0/3.0)
|   |   credit_history = all paid: bad (9.0/1.0)
|   |   credit_history = existing paid
|   |   |   other_parties = none
|   |   |   |   savings_status = <100
|   |   |   |   |   existing_credits <= 1
|   |   |   |   |   |   purpose = new car: bad (17.0/4.0)
|   |   |   |   |   |   purpose = used car: good (3.0/1.0)
|   |   |   |   |   |   purpose = furniture/equipment: good (22.0/11.0)
|   |   |   |   |   |   purpose = radio/tv: good (18.0/8.0)
|   |   |   |   |   |   purpose = domestic appliance: bad (2.0)
|   |   |   |   |   |   purpose = repairs: bad (1.0)
|   |   |   |   |   |   purpose = education: bad (5.0/1.0)
|   |   |   |   |   |   purpose = vacation: bad (0.0)
|   |   |   |   |   |   purpose = retraining: bad (0.0)
|   |   |   |   |   |   purpose = business: good (3.0/1.0)
|   |   |   |   |   |   purpose = other: bad (0.0)
|   |   |   |   |   existing_credits > 1: bad (5.0)
|   |   |   |   savings_status = 100<=X<500: bad (8.0/3.0)
|   |   |   |   savings_status = 500<=X<1000: good (1.0)
|   |   |   |   savings_status = >=1000: good (2.0)
|   |   |   |   savings_status = no known savings
|   |   |   |   |   job = unemp/unskilled non res: bad (0.0)
|   |   |   |   |   job = unskilled resident: good (2.0)
|   |   |   |   |   job = skilled
|   |   |   |   |   |   own_telephone = none: bad (4.0)
|   |   |   |   |   |   own_telephone = yes: good (3.0/1.0)
|   |   |   |   |   job = high qualif/self emp/mgmt: bad (3.0/1.0)
|   |   |   other_parties = co applicant: good (4.0/2.0)
|   |   |   other_parties = guarantor: good (8.0/1.0)
|   |   credit_history = delayed previously: bad (7.0/2.0)
|   |   credit_history = critical/other existing credit: good (38.0/10.0)
|   foreign_worker = no: good (12.0/2.0)
checking_status = 0<=X<200
|   other_parties = none
|   |   credit_history = no credits/all paid
|   |   |   other_payment_plans = bank: good (2.0/1.0)
|   |   |   other_payment_plans = stores: bad (0.0)
|   |   |   other_payment_plans = none: bad (7.0)
|   |   credit_history = all paid: bad (10.0/4.0)
|   |   credit_history = existing paid
|   |   |   credit_amount <= 8858: good (70.0/21.0)
|   |   |   credit_amount > 8858: bad (8.0)
|   |   credit_history = delayed previously: good (25.0/6.0)
|   |   credit_history = critical/other existing credit: good (26.0/7.0)
|   other_parties = co applicant: bad (7.0/1.0)
|   other_parties = guarantor: good (18.0/4.0)
checking_status = >=200: good (44.0/9.0)
checking_status = no checking
|   other_payment_plans = bank: good (30.0/10.0)
|   other_payment_plans = stores: good (12.0/2.0)
|   other_payment_plans = none
|   |   credit_history = no credits/all paid: good (4.0)
|   |   credit_history = all paid: good (1.0)
|   |   credit_history = existing paid
|   |   |   existing_credits <= 1: good (92.0/7.0)
|   |   |   existing_credits > 1
|   |   |   |   installment_commitment <= 2: bad (4.0/1.0)
|   |   |   |   installment_commitment > 2: good (5.0)
|   |   credit_history = delayed previously: good (22.0/6.0)
|   |   credit_history = critical/other existing credit: good (92.0/3.0)

Number of Leaves  :     47

Size of the tree :      64

Time taken to build model: 0.49 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          725               72.5    %
Incorrectly Classified Instances        275               27.5    %
Kappa statistic                           0.2786
Mean absolute error                       0.3331
Root mean squared error                   0.4562
Relative absolute error                  79.2826 %
Root relative squared error              99.5538 %
Total Number of Instances              1000
```

12. (Extra Credit): How can you convert a Decision Trees into "if-then-else rules". Make up your own small Decision Tree consisting of 2-3 levels and convert it into a set of rules. There also exist different classifiers that output the model in the form of rules - one such classifier in Weka is rules.PART, train this model and report the set of rules obtained. Sometimes just one attribute can be good enough in making the decision, yes, just one ! Can you predict what attribute that might be in this dataset? OneR classifier uses a single attribute to make decisions (it chooses the attribute based on minimum error). Report the

rule obtained by training a one R classifier. Rank the performance of j48, PART and oneR.

In weka, rules.PART is one of the classifier which converts the decision trees into "IF-THEN-ELSE" rules.

<u>Converting Decision trees into "IF-THEN-ELSE" rules using rules.PART classifier:-</u>

<u>PART decision list</u>
outlook = overcast: yes (4.0)

windy = TRUE: no (4.0/1.0)

outlook = sunny: no (3.0/1.0)

: yes (3.0)

Number of Rules : 4

Yes, sometimes just one attribute can be good enough in making the decision. In this dataset (Weather), Single attribute for making the decision is **"outlook"**

outlook:
sunny -> no
overcast -> yes
rainy -> yes
(10/14 instances correct)

With respect to the **time**, the oneR classifier has higher ranking and J48 is in 2nd place and PART gets 3rd place.

	J48	PART	oneR
TIME (sec)	0.12	0.14	0.04
RANK	II	III	I

But if you consider the **accuracy,** The J48 classifier has higher ranking, PART gets second place and oneR gets lst place

	J48	PART	oneR
ACCURACY (%)	70.5	70.2%	66.8%
RANK	I	II	III

BTW, If you find any mistakes, please mail me, and I'll correct them ASAP and all Credits goes to me....☺ :D :P