2.Data Pre-Processing:

For this question I worked with the suggested dataset, mainly because it needed a good amount of processing in order to be usable, and that wasn't the same for many of the ones that I came across.

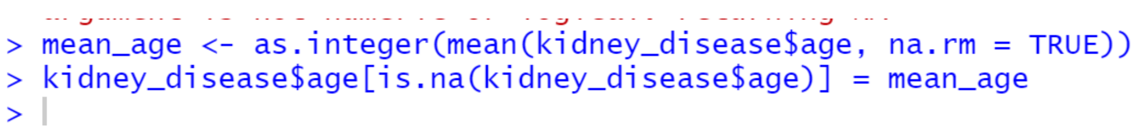
Here is how the chronic kidney disease dataset was described on Kaggle:

The data was taken over a 2-month period in India with 25 features (eg, red blood cell count, white blood cell count, etc). The target is the 'classification', which is either 'ckd' or 'notckd' - ckd=chronic kidney disease.

Here is a full list of each column in the dataset:

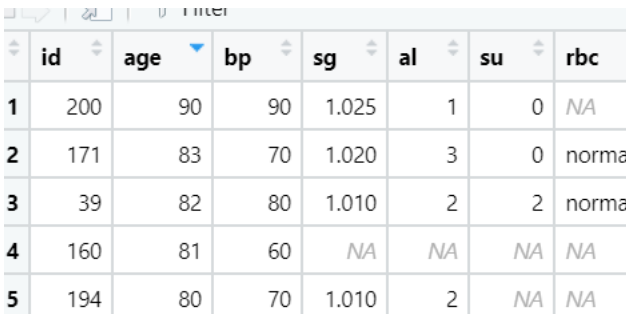
age - age	sod - sodium
bp - blood pressure	pot - potassium
sg - specific gravity	hemo - hemoglobin
al - albumin	pcv - packed cell volume
su - sugar	wc - white blood cell count
rbc - red blood cells	rc - red blood cell count
pc - pus cell	htn - hypertension
pcc - pus cell clumps	dm - diabetes mellitus
ba - bacteria	cad - coronary artery disease
bgr - blood glucose random	appet - appetite
bu - blood urea	pe - pedal edema
sc - serum creatinine	ane - anemia
	class - classification

I worked from left to right. I got the mean of all ages after removing the 'NA's' and them with the mean age of all participants.



```
> mean_age <- as.integer(mean(kidney_disease$age, na.rm = TRUE))
> kidney_disease$age[is.na(kidney_disease$age)] = mean_age
> 
```

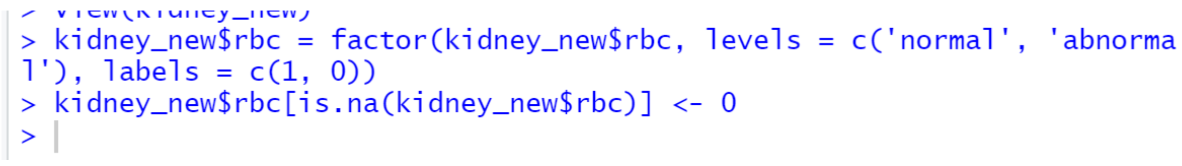
I repeated the same steps for each of the following columns until I got to 'rbc' (red blood cells).



	id	age	bp	sg	al	su	rbc
1	200	90	90	1.025	1	0	NA
2	171	83	70	1.020	3	0	norma
3	39	82	80	1.010	2	2	norma
4	160	81	60	NA	NA	NA	NA
5	194	80	70	1.010	2	NA	NA

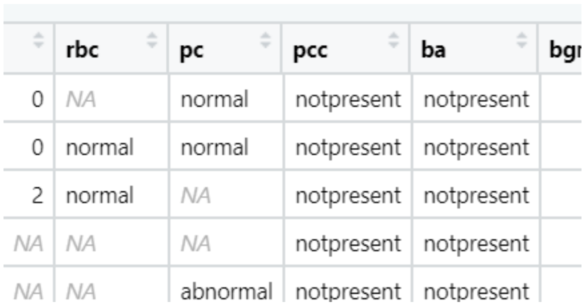
In 'rbc' we meet 'characters' for the first time. My approach here was what was presented to us in class – that in order to make the code machine readable the bivalent values should be changed to either a numerical '1 and 0' or a logical 'True and False'.

I assigned 'normal' to '1' and 'abnormal' to a '0' and changed the values for the whole column, and subsequently assigned 'NA' to '0'.



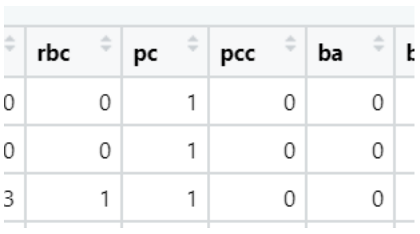
```
> View(kidney_new)
> kidney_new$rbc = factor(kidney_new$rbc, levels = c('normal', 'abnorma
l'), labels = c(1, 0))
> kidney_new$rbc[is.na(kidney_new$rbc)] <- 0
> 
```

This process was repeated for another four columns, the only difference being the nature of the character strings being replaced:



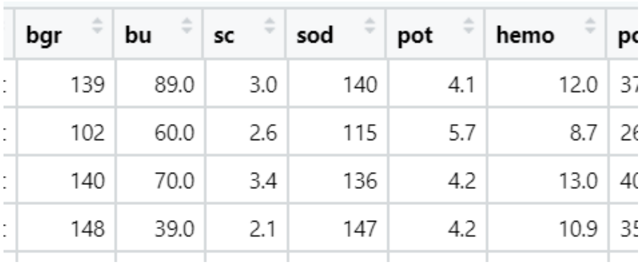
	rbc	pc	pcc	ba	bg
0	NA	normal	notpresent	notpresent	
0	normal	normal	notpresent	notpresent	
2	normal	NA	notpresent	notpresent	
NA	NA	NA	notpresent	notpresent	
NA	NA	abnormal	notpresent	notpresent	

the 'pcc' and 'ba' columns (0,1) columns were 'notpresent' and 'present' respectively.



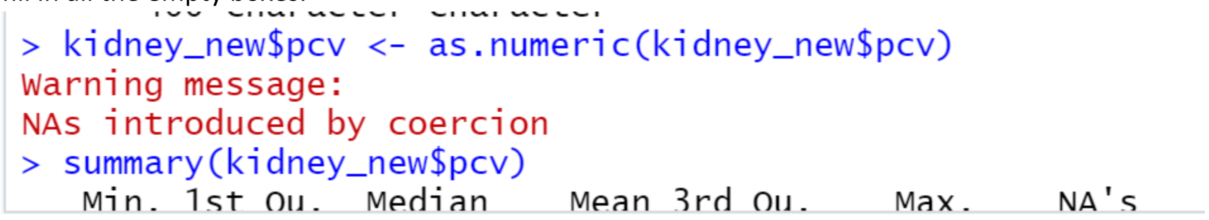
	rbc	pc	pcc	ba	b
0	0	1	0	0	
0	0	1	0	0	
3	1	1	0	0	

The next six columns were treated the same way as the first five, by calculating the mean value of each column and replacing the 'NA' values with the result.



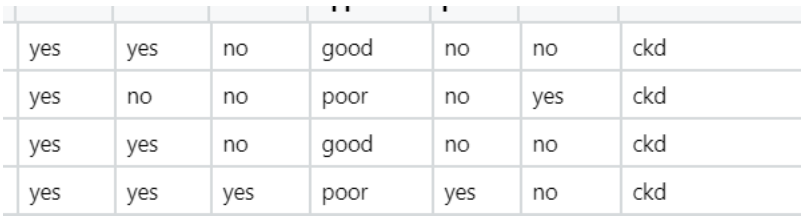
bgr	bu	sc	sod	pot	hemo	pc
139	89.0	3.0	140	4.1	12.0	37
102	60.0	2.6	115	5.7	8.7	26
140	70.0	3.4	136	4.2	13.0	40
148	39.0	2.1	147	4.2	10.9	35

Interestingly, the 'pcv' column threw me a curveball, as though it was a list of integer values they were listed as being 'characters. I assume that was because they were entered into the dataset from user input. This meant I had to change all the values to 'numeric' before I could find the mean and so fill in all the empty boxes.



```
> kidney_new$pcv <- as.numeric(kidney_new$pcv)
Warning message:
NAs introduced by coercion
> summary(kidney_new$pcv)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
```

The next two columns were numerical and the remaining eight were all characters – all of them bivalent in nature. They had different permutations but were all either true or false in nature – 'yes/no', 'good/poor' and 'ckd/notckd'.



yes	yes	no	good	no	no	ckd
yes	no	no	poor	no	yes	ckd
yes	yes	no	good	no	no	ckd
yes	yes	yes	poor	yes	no	ckd

All columns were assigned ones and zeros and NA's were removed.

By the end of this process I had a dataset that was machine ready with all NA values removed and all columns being 'numeric' in form.