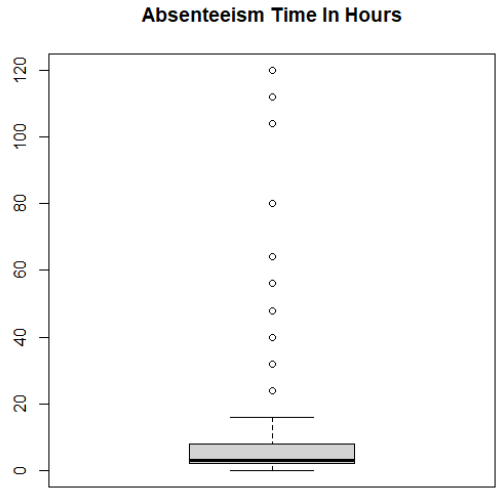
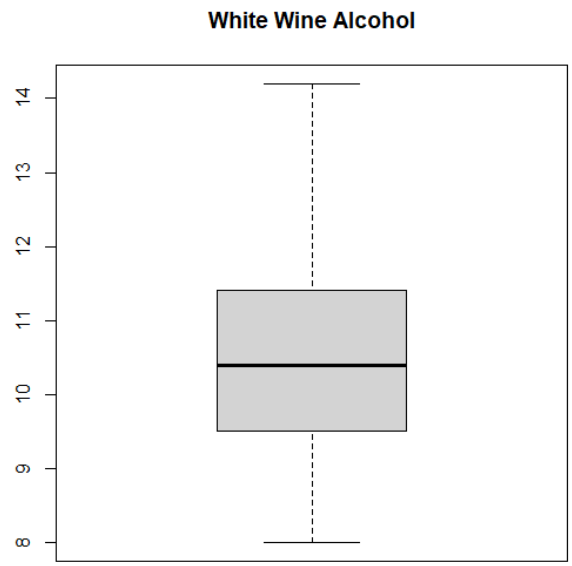
Ryan Kaplan

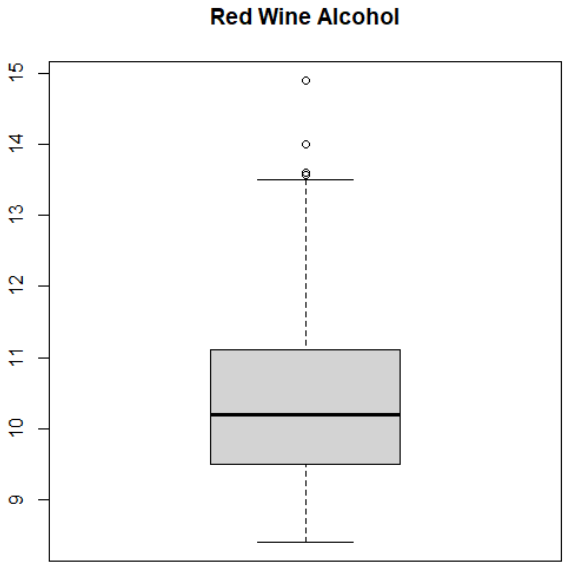
1) The first dataset I examined was the work absenteeism dataset. For cleaning, when I tried loading it in, the entire dataset showed up as a single column, so I had to separate columns using semicolons. There were no missing values, which makes things a little easier. When I looked at the summary of each column, other than the target variable, the number of absentee hours for each employee, the ranges were reasonable. For all but one, the min and max values were not orders of magnitude away from the median, however the actual target variable was quite skewed. The median absentee time is 3 hours, with a first and third quartile of 2 and 8 hours, respectively, however the max was 120 hours, or a whopping 5 days worth of absenteeism. The following is the box plot for absentee hours:



While the minimum number of absentee hours was 0, this is likely due to rounding down, and so should not be considered an outlier. Removing examples with greater than 20 absentee hours leaves us with 696 data points, which leaves us with more than enough examples for meaningful analysis. I think that a regression approach seems

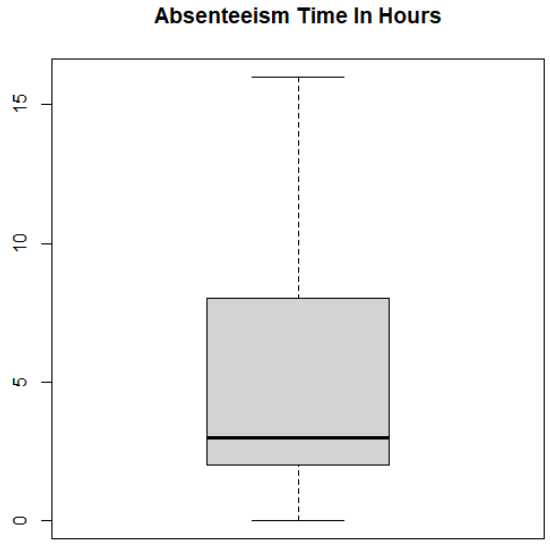
reasonable, and I'd also like to use knn, to predict the target variable, the number of absentee hours. It'll be interesting to compare their performances on this same task. The second dataset I picked was wine quality. The summaries for both the red and white wine again looked good, and it was again helpful that there are no missing variables. With the exception of total sulfur dioxide, the max and min values aren't too far from the mean. Since some wines do have high sulfur dioxide, I judged this to not be an outlier. The two datasets had many similar variable distributions, as one would expect. For example, the following is the distributions of alcohol content for red and white wine, respectively:





The two are nearly the same, though of course there are small exceptions, such as the maximum alcohol content being greater for red than white wine, albeit with only a single example of 15.I think this also works well for regression and knn, as its another example of a number being predicted, but with a relatively small range of possible values.

2) For absenteeism the target variable is, fittingly, the number of absentee hours for each employee. After cleaning, we are left with the following distribution of hours:

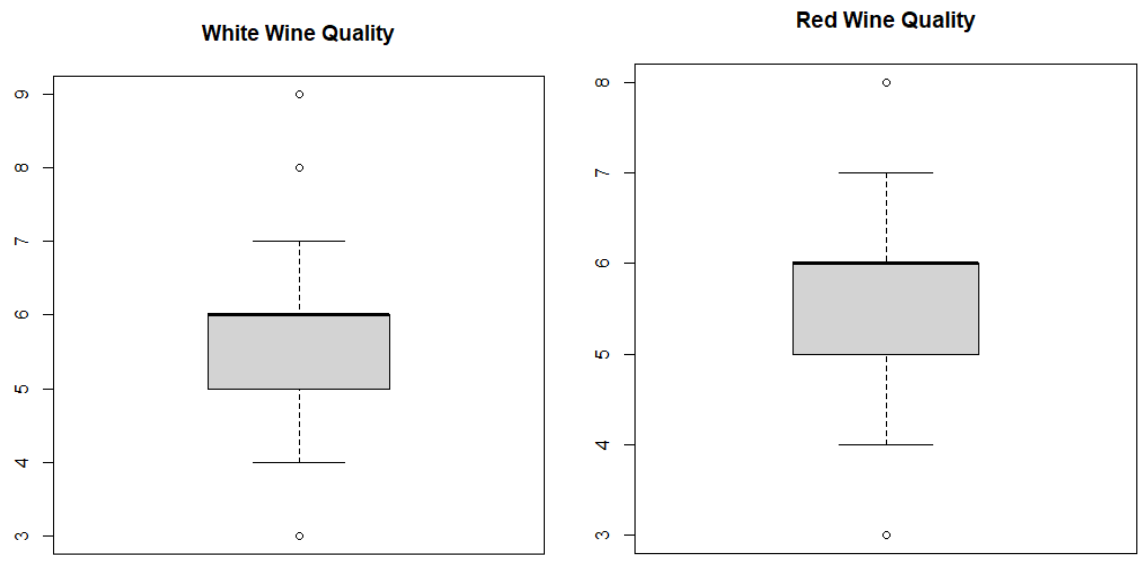


A twenty point range shouldn't be too difficult to predict, and most of the results are within a smaller range, between 2 and 8. We can do linear regression on this data, and the results turned out pretty well. I chose to include all possible variables, as they were all numeric, so no conversion was necessary, and I thought they might all be relevant to predicting absenteeism, with the possible exception of employee ID, as I was unsure if how these IDs were assigned. Other than that, the connections seemed logically straightforward. For example, it should be the case that having children or pets increases the likelihood, or duration of, absenteeism due to the task necessary for their care, and the reason for absence surely has an impact, with more serious reasons resulting in a greater degree of absenteeism, and vice versa. The following are the coefficients of each variable:

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       47.846976  18.482650   2.589  0.00984 **
ID                -0.052500   0.015287  -3.434  0.00063 ***
absence_reason    -0.170631   0.016411 -10.398  < 2e-16 ***
month             -0.002564   0.043253  -0.059  0.95274
day               -0.113622   0.080009  -1.420  0.15604
season            -0.056948   0.117690  -0.484  0.62863
transport_expense  0.007068   0.002227   3.173  0.00158 **
distance_to_work  -0.017594   0.011893  -1.479  0.13950
service_time      -0.125118   0.048934  -2.557  0.01078 *
age               -0.009437   0.029508  -0.320  0.74920
average_workload   0.002976   0.003208   0.928  0.35384
hit_target        -0.005369   0.035488  -0.151  0.87979
discipline_failure -8.768753  0.596853 -14.692  < 2e-16 ***
education         -0.101146   0.219600  -0.461  0.64524
number_of_children  0.337613  0.118303   2.854  0.00445 **
drinker             0.608547  0.370359   1.643  0.10082
smoker              1.061413  0.485227   2.187  0.02905 *
num_pets           -0.316233  0.111414  -2.838  0.00467 **
weight              0.251996  0.114391   2.203  0.02794 *
height             -0.231182  0.104287  -2.217  0.02697 *
BMI                -0.678025  0.330443  -2.052  0.04057 *
```

The two most important points to look at here are the estimate of coefficients, and the p-value for each variable. Because linear regression works by multiplying each variable by a coefficient and summing them, a coefficient with a very low value likely doesn't matter very much. For example, if a variable that ranged from 0 to 10 had a coefficient of 0.0001, that variable would have almost no impact on the final result.While coefficient is useful, the variable we really care about here is our p-value, in the Pr column. This represents the probability that this predictor variable isn't meaningful for our regression. So, a high value such as 0.5 would mean there's only a 50% chance that the predictor variable matters. Typically, we look for p-values of less than 0.05, which would represent a 95% chance that the predictor is useful here. We can see right away that some of the variables matter much more than others. For example, I would not have guessed that transport expense would matter much more than the actual distance to work, however we can see it's very important, with a 99.9% chance of mattering, whereas distance to work has only an 86% chance. Using regression, the median prediction was only 0.44 hours off, with first and third quartiles of negative and positive 1.6, respectively. While

the worst errors, in the positive and negative direction, were 12.9 and 8.7 hours off, with those quartile results I can confidently say that regression does a good job of predicting the number of absentee hours. I chose to do knn first using every variable, and then only the ones deemed significant by the regression algorithm. Using 22 nearest neighbors, including all variables gives an accuracy of 41.9%, which is pretty good considering this is based on the average result of neighbors, not as fine-tuned as regression can be. If we only include variables with a p-value of less than 0.05, we get an accuracy of 45.2%, which is clearly superior, but not as different as one might initially think. For the wine datasets, the target variable is the wine's quality score. We have the following distribution:



Again we see very similar distributions, and since theyre close together it should be relatively easy to predict using regression. Again, all the variables are numeric and seem relevant, so we'll include them all. I may not be a wine expert, but it makes sense

to me that ph would have an influence on quality, for example. The following are the coefficients for the red and white wine data:

Red
Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	2.197e+01	2.119e+01	1.036	0.3002	
fixed_acidity	2.499e-02	2.595e-02	0.963	0.3357	
volatile_acidity	-1.084e+00	1.211e-01	-8.948	< 2e-16	***
citric_acid	-1.826e-01	1.472e-01	-1.240	0.2150	
residual_sugar	1.633e-02	1.500e-02	1.089	0.2765	
chlorides	-1.874e+00	4.193e-01	-4.470	8.37e-06	***
free_sulfur_dioxide	4.361e-03	2.171e-03	2.009	0.0447	*
total_sulfur_dioxide	-3.265e-03	7.287e-04	-4.480	8.00e-06	***
density	-1.788e+01	2.163e+01	-0.827	0.4086	
ph	-4.137e-01	1.916e-01	-2.159	0.0310	*
sulphates	9.163e-01	1.143e-01	8.014	2.13e-15	***
alcohol	2.762e-01	2.648e-02	10.429	< 2e-16	***

White
Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	1.502e+02	1.880e+01	7.987	1.71e-15	***
fixed_acidity	6.552e-02	2.087e-02	3.139	0.00171	**
volatile_acidity	-1.863e+00	1.138e-01	-16.373	< 2e-16	***
citric_acid	2.209e-02	9.577e-02	0.231	0.81759	
residual_sugar	8.148e-02	7.527e-03	10.825	< 2e-16	***
chlorides	-2.473e-01	5.465e-01	-0.452	0.65097	
free_sulfur_dioxide	3.733e-03	8.441e-04	4.422	9.99e-06	***
total_sulfur_dioxide	-2.857e-04	3.781e-04	-0.756	0.44979	
density	-1.503e+02	1.907e+01	-7.879	4.04e-15	***
ph	6.863e-01	1.054e-01	6.513	8.10e-11	***
sulphates	6.315e-01	1.004e-01	6.291	3.44e-10	***
alcohol	1.935e-01	2.422e-02	7.988	1.70e-15	***

Perhaps ignorantly,I was surprised to see that there were significant differences in which variables were relevant to the different types of wine. For example, we see that fixed acidity has a 0.1% chance to not be relevant for white wine, whereas for red wine it only has a 67% chance of mattering. Some of them are more or less relevant for both of them, such as alcohol content being incredibly important for both, and citric acid mattering less, but overall I was surprised to find large differenced in p-value for any of

them. We also see more of them mattering than in the absenteeism dataset, which makes sense given that there are less variables, and are all relevant to the quality of the wine. I suppose its a good thing that non-useful information is not included, such as the month the vintner was born. For red wine, the median error for our prediction was incorrect by only 0.04, with 1st and third quartiles of negative 0.36, and positive 0.45, which considering the score is out of 10 is a great accuracy score. The maximum and minimum error were only negative 2.68 and 2.02, so all of our predictions were within a pretty tight range of true. Similarly, the median prediction for white wine was only off by 0.0379. The other quartiles reflect this as well, with a first and third quartile of negative and positive 0.4, with a max and min difference from true of negative 3.8 and positive 3.1. Again, regression seems to do very well here. For knn on the red wine dataset, using every variable and 33 neighbors, we get an accuracy of 58%, which again is not bad for guessing a number up to 10. Interestingly, if we remove the less significant variables our accuracy slightly worsens to 57.58%. For knn on the white data, our accuracy is a little worse, at 41.1%, which is still not too bad considering its knn. If we again remove the less relevant variables, this time our accuracy is greatly improved to 47.8%. Looking back at the removed variables for the red data, their p-values are indeed about 0.05, but still no worse than a 60% chance of being relevant, so I suppose it makes sense that they’re useful for predicting wine quality. For white wine however, for example, citric acid has an 81% chance to not be useful, so its definitely correct to remove it.

3) As mentioned previously, I think both models performed well. For regression, using the absenteeism dataset the majority of our predictions were within two hours of the true result, and for both wine sets, the majority of our predictions were very close, and none of them were 4 or more away from true. For knn, we got initial accuracies of 41.9 for the absenteeism set, with red and white wine accuracies of 58 and 41.1, respectively. Using the regression coefficients to choose which variables to include, redoing knn gives us an absenteeism accuracy of 45.2, plus red and white wine accuracies of 57.58 and 47.8.While these accuracies are not too bad for knn, considering its predicting a number rather than a category here, a task for which its better equipped, it does show a limitation of the knn algorithm. Since its a “lazy” algorithm, meaning that its training is just storing the data so it can just perform the output calculation after receiving input, we have few ways to optimize its results. The main ways include picking the number of neighbors to use, calculating distance differently, and most importantly, picking which variables to use, however we don’t have a good way to determine which variables should and should not be included. Here, I used regression to determine which variables were most relevant, and was usually able to improve our knn performance by limiting our determination to these variables, however this is using an outside method. Using only knn, we’re limited in our ability to improve it other than essentially guess and check methods, outside a few rule of thumb methods, like setting the number of clusters to the square root of the size of our train set. Of course, its generally faster, easier to implement and intuitively understand, etc. This is not to say that knn is without advantage, however for hard problems, particularly outside of classification, an actual learning method will usually be superior to a naive one.