Homework #2

Rufus Petrie

Professors Magnolfi and Sullivan

Machine Learning

9 October, 2018

Note: Because the code to clean the data outputted a lot of useless information, I have silenced it in this file. The code is still visible in the script and Rmarkdown files included in this folder.

Question 3

```
## [1] "The adjusted r-squared equals: 0.250881198078531"
```

```
## [1] "The test MSE equals: 2.18263481155721"
```

Question 4

```
## [1] "The adjusted r-squared equals: 0.354743705804456"
```

```
## [1] "The test MSE equals: 1.84557157765079"
```

Question 5

```
## Reordering variables and trying again:
```

```
## [1] "The test MSE for the model selected by adjusted r-squared equals: 2.34793806783376"
```

```
## [1] "The test MSE for the model selected by BIC equals: 2.91260137055951"
```

Both of these models perform significantly worse than the simple linear model. This is likely because by selecting the model that has the best in-sample adjusted r-squared or BIC, we are overfitting the model to the dataset.

Question 6

	lambda=0	lambda=1	lambda=2
ridge	2.182637	2.290554	2.394291
lasso	2.182637	2.961425	2.961425

Above is a table describing the test MSE for the ridge/lasso models with varying penalty parameters. Notice that the predictive performance of the model decreases as the penalty increases. This is likely because the simple linear model is underfitted, i.e. adding more interactions or squared terms could help the predictive performance.

Question 7

```
## [1] "The test MSE for the 10-fold CV ridge equals: 2.18781270131042"
```

```
## [1] "The test MSE for the 10-fold CV lasso equals: 2.18781270131042"
```

As we can see, the ridge and lasso models have comparable performance to the simple linear model. As I mentioned before, this is likely because the simple linear model is underfitted, meaning that it would achieve better predictive performance if it had more interaction or polynomial terms. We can see this because of the models predicted so far, the polynomial model with no selection had the lowest test MSE (albeit this was only predicted using a 50/50 test train split). However, the ridge and lasso models still perform better than the selected polynomial model, which was likely overfitted to the training set because of its poor out of sample performance.

Question 8

Of the linear models proposed, I believe that simple regression is most appropriate. Both the ridge and the lasso models had very low penalty parameters, which resulted in the test MSE values being close to the linear regression test MSE. Although the simple regression is only evaluted on a test/train split (it would be nice to do k-fold cv for it), it appears that a linear model may be underspecified, i.e. it is not flexible enough. Therefore, it might be possible to find a nonlinear model that performs better than the linear regression, but we would need to test some more models.

Question 9

10/90 train/test split

```
## [1] "Linear model adjusted r-squared equals: 0.249997892410143"
```

```
## [1] "Linear model test MSE equals: 2.21993828514083"
```

```
## [1] "Polynomials adjusted r-squared equals: 0.379888092966545"
```

```
## [1] "Polynomial test MSE equals: 2.0777987389252"
```

```
## Reordering variables and trying again:
```

```
## [1] "Polynomial selected with adjusted r-squared test MSE equals: 2.25607028800445"
```

```
## [1] "Polynomial selected with BIC test MSE equals: 2.92363590158425"
```

	lambda=0	lambda=1	lambda=2
ridge	2.219944	2.303168	2.404886
lasso	2.219944	2.967444	2.967444

```
## [1] "10-fold CV ridge test MSE equals: 2.22000255764025"
```

```
## [1] "10-fold CV lasso test MSE equals: 2.22000255764025"
```

2/98 train/test split

```
## [1] "Linear model adjusted r-squared equals: 0.2783963370724"
## [1] "Linear model test MSE equals: 2.2593455112084"
## [1] "Polynomials adjusted r-squared equals: 0.352802301141731"
## [1] "Polynomial test MSE equals: 3.14317250827349"
## Reordering variables and trying again:
## [1] "Polynomial selected with adjusted r-squared test MSE equals: 2.43135857300266"
## [1] "Polynomial selected with BIC test MSE equals: 2.95720060591436"
```

	lambda=0	lambda=1	lambda=2
ridge	2.259337	2.322470	2.410801
lasso	2.259337	2.969029	2.969029

```
## [1] "10-fold CV ridge test MSE equals: 2.27078684237481"
## [1] "10-fold CV lasso test MSE equals: 2.27078684237481"
```

50/50 train/test split, noise

```
## [1] "Linear model adjusted r-squared equals: 0.256754482788848"
## [1] "Linear model test MSE equals: 2.16643461118809"
## [1] "Polynomials adjusted r-squared equals: 0.363049436577785"
## [1] "Polynomial test MSE equals: 1.86310758003603"
## Reordering variables and trying again:
## [1] "Polynomial selected with adjusted r-squared test MSE equals: 1.85979367990186"
## [1] "Polynomial selected with BIC test MSE equals: 2.861300034759"
```

	lambda=0	lambda=1	lambda=2
ridge	2.156592	2.196832	2.262546
lasso	2.156592	2.915061	2.915061

```
## [1] "10-fold CV ridge test MSE equals: 2.15550234589723"
## [1] "10-fold CV lasso test MSE equals: 2.15550234589723"
```

10/90 train/test split, noise

```
## [1] "Linear model adjusted r-squared equals: 0.263728129138799"
## [1] "Linear model test MSE equals: 2.22891005441869"
## [1] "Polynomials adjusted r-squared equals: 0.374477894448882"
## [1] "Polynomial test MSE equals: 2.07380000464961"
## Reordering variables and trying again:
## [1] "Polynomial selected with adjusted r-squared test MSE equals: 2.11893435059457"
## [1] "Polynomial selected with BIC test MSE equals: 2.86505693055661"
```

	lambda=0	lambda=1	lambda=2
ridge	2.222697	2.270818	2.334241
lasso	2.222697	2.969441	2.969441

```
## [1] "10-fold CV ridge test MSE equals: 2.22978497935273"
## [1] "10-fold CV lasso test MSE equals: 2.22978497935273"
```

2/98 train/test split, noise

```
## [1] "Linear model adjusted r-squared equals: 0.248211610110318"
## [1] "Linear model test MSE equals: 2.4077223469301"
## [1] "Polynomials adjusted r-squared equals: 0.48172899081262"
## [1] "Polynomial test MSE equals: 3.79064038085513"
## Reordering variables and trying again:
## [1] "Polynomial selected with adjusted r-squared test MSE equals: 2.68688529501788"
## [1] "Polynomial selected with BIC test MSE equals: 3.01895446452784"
```

	lambda=0	lambda=1	lambda=2
ridge	2.349064	2.354667	2.421064
lasso	2.349064	2.960923	2.960923

```
## [1] "10-fold CV ridge test MSE equals: 2.35180254201385"
## [1] "10-fold CV lasso test MSE equals: 2.35180254201385"
```

In general, the test MSE of the models decreases as we allocate more data to the test set and less to the training set. This is likely because the variance of our modle increases as we decrease the size of the training sample, which result in increasing test MSE. Furthermore, the test MSE generally decreases as we add more noise. With an equal train/test split, it looks like the noise can actually improve performance (to be expected because with mean 0, estimates will be unbiased), but the noise really hurts test MSE when the train/test split becomes uneven because the errors don't necessarily cancel out on average.

Question 10

```
## Subset selection object
## Call: regsubsets.formula(y ~ x, data = datam[train, ], method = "backward")
## 6 Variables  (and intercept)
##                      Forced in Forced out
## xHub                     FALSE      FALSE
## xVacation_spot           FALSE      FALSE
## xMean_income             FALSE      FALSE
## xSlot_controlled         FALSE      FALSE
## xaverage_distance_m      FALSE      FALSE
## xmarket_size             FALSE      FALSE
## 1 subsets of each size up to 6
## Selection Algorithm: backward
##          xHub xVacation_spot xMean_income xSlot_controlled
## 1  ( 1 ) " "  " "            " "          " "
## 2  ( 1 ) "*"  " "            " "          " "
## 3  ( 1 ) "*"  " "            " "          " "
## 4  ( 1 ) "*"  "*"            " "          " "
## 5  ( 1 ) "*"  "*"            "*"          " "
## 6  ( 1 ) "*"  "*"            "*"          "*"
##          xaverage_distance_m xmarket_size
## 1  ( 1 ) "*"                 " "
## 2  ( 1 ) "*"                 " "
## 3  ( 1 ) "*"                 "*"
## 4  ( 1 ) "*"                 "*"
## 5  ( 1 ) "*"                 "*"
## 6  ( 1 ) "*"                 "*"
```

```
## [1] 0.1817702 0.2339740 0.2441734 0.2500202 0.2514911 0.2523467
```

From the model selection procedure, we can see that the three most important variables for determining the number of carriers are the average distance, hub, and market size variables. After including these three variables, each additional variable included reduces the variation in the data by less than 1%.

	x
(Intercept)	1.5356668
xHub	0.9541959
xVacation_spot	-0.2174117
xMean_income	0.0000242
xSlot_controlled	-0.2060253
xaverage_distance_m	0.0008743
xmarket_size	0.0000001

Notice that an airport being a hub raises its predicted number of carriers by almost 1. It's harder to interpret the market size and distance variables, but the fact that these are positive seems intuitive enough.