COM S 474/574 Spring 2020
Introduction to Machine Learning

Homework 2

1 Directions:

- **Due: Thursday February 27, 2020 at 10pm.** Late submissions will be accepted for 24 hours after that time, with a 15% penalty.
- Upload the homework to Canvas as a pdf file. Answers to problems 1-4 can be handwritten, but writing must be neat and the scan should be high-quality image. Other responses should be typed or computer generated.
- Any non-administrative questions must be asked in office hours or (if a brief response is sufficient) Piazza.

2 Problems

In this homework, we will focus on using nearest neighbor methods and logistic regression to classify (predict a discrete-valued feature y, such as $y \in \{0,1\}$).

Problem 1. [5 points] Suppose you are predicting feature y using feature x with logistic regression, and x is measured in kilometers. After fitting, you get coefficients $\beta_0 = 1.24$ and $\beta_1 = -3.74$. Thus, your model is

Prob($y = 1|x) = \frac{e^{1.24-3.74x}}{1+e^{1.24-3.74x}}$

Suppose our friend Sammie has an innate fear of the metric system, starts with the same data set, converts the x values to miles, does not change y values, and then fits. What will Sammie's β_0 and β_1 be? $\beta 0 = 1.24$ $\beta 1 = 6.01895$

Problem 2. [15 points] Book problem Chapter 4, #4 "When the number of features ..."

(a) Suppose that we have a set of observations, each with measurements on p = 1 feature,X. We assume that X is uniformly (evenly) distributed on [0,1]. Associated with each observation is a response value. Suppose that we wish to predict a test observation's response using only observations that are within 10% of the range of X closest to that test observation. For instance, in order to predict the response for a test observation with X =0 .6, we will use observations in the range [0.55,0.65]. On average, what fraction of the available observations will we use to make the prediction? 0.1

(b) Now suppose that we have a set of observations, each with measurements on p = 2 features,X1 and X2. We assume that (X1,X2) are uniformly distributed on [0,1]×[0,1]. We wish to predict a test observation's response using only observations that are within 10% of the range of X1 and within 10% of the range of X2 closest to that test observation. For instance, in order to predict the response for a test observation with X1 =0 .6 and X2 =0 .35, we will use observations in the range [0.55,0.65] for X1 and in the range [0.3,0.4] for X2. On average, what fraction of the available observations will we use to make the prediction? 0.01

(c) Now suppose that we have a set of observations on p = 100 features. Again the observations are uniformly distributed on each feature, and again each feature ranges in value from 0 to 1. We wish to predict a test observation's response using observations within the 10% of each feature's range that is closest to that test

observation. What fraction of the available observations will we use to make the prediction?
1e-100

(d) Using your answers to parts (a)–(c), argue that a drawback of KNN when p is large is that there are very few training observations "near" any given test observation.

As the dimensions increase, the volume inscribed in the hypercube gets to be a smaller and smaller volume relative to the hypercube eventually going to a volume of 0 in infinite dimensions. Because of this, the area being searched will contain fewer and fewer observations as the dimesons increase. This is assuming the inscribed edge length remains constant.

(d) Now suppose that we wish to make a prediction for a test observation by creating a p-dimensional hypercube centered around the test observation that contains, on average, 10% of the training observations. For p =1 ,2, and 100, what is the length of each side of the hypercube? Comment on your answer.
X=edge length
X^1=0.1 X = 0.1
X^2=0.1 X = 0.316
X^100=0.1 X = 0.977

To cover 10% of the p feature range, one needs to obtain a greater percentage of the observations as p increases. If the amount of available observations is fixed, then overfitting occurs if we keep adding dimensions.

Problem 3. [10 points] Book problem Chapter 4, #6 "Suppose we collect data ..."

6. Suppose we collect data for a group of students in a statistics class with variables X1 =hours studied, X2 =undergrad GPA, and Y = receive an A. We fit a logistic regression and produce estimated coefficient, $\hat{\beta}0 = -6$, $\hat{\beta}1 = 0.05$, $\hat{\beta}2 = 1$.

(a) Estimate the probability that a student who studies for 40h and has an undergrad GPA of 3.5 gets an A in the class.

(e^(-6 + 0.05(40) + 1(3.5)))/(1+(e^(-6 + 0.05(40) + 1(3.5)))) =0.37754

(b) How many hours would the student in part (a) need to study to have a 50% chance of getting an A in the class?

50 Hours

Problem 4. [5 points] Book problem Chapter 4, #8 "Suppose that we take a data set ..."
Note: For the following problem, instead of reporting training/testing loss, for simplicity you will be asked to report accuracy. Accuracy is the percentage of samples that were correctly labeled as $y = 0$ *or* $y = 1$.

Suppose that we take a data set, divide it into equally-sized training and test sets, and then try out two different classification procedures. First we use logistic regression and get an error rate of 20% on the training data and **30% on the test data**. Next we use 1-nearest neighbors (i.e. K = 1) and get an average error rate (averaged over both test and training data sets) of 18%. Based on these results, which method should we prefer to use for classification of new observations? Why?

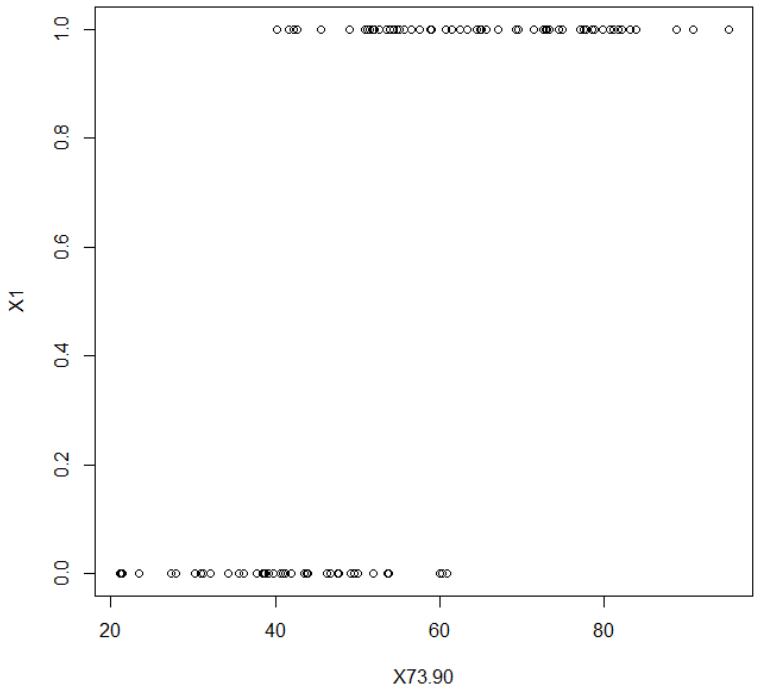
We should use logistic regression since the accuracy on the test data is better.

For the 1-nearest neighbors, assuming K=1 fits the training data perfectly, then it would have 64% accuracy for the testing data.

Logistic regression has a 70% accuracy on the test data.

Problem 5. [65 points]

A. Download the data sets HW2train.csv and HW2test.csv from Canvas. In both files, the first column is a binary-valued feature *y*. The second column is a continuous-valued feature *x*. Make a scatter-plot of the data-set HW2train with *y* values on the vertical axis, *x* values on the horizontal axis.



B. Fit a logistic model to predict y. Use the whole data set HW2train. If you are using Python, you can use https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.

- The argument 'penalty' is whether we want to penalize the coefficients. For this assignment, we will use 'penalty'=none.
- Set the argument 'fitintercept'=True to add a β_0 (provided that we do not include a column of ones when we call the .fit() function). The "intercept" β_0 will be stored as .intercept while feature coefficients $\beta_1,\beta_2,...$ are stored in .coef_

(1) Report the β_0 and β_1 values you obtain.

β_0 = -9.05870499

β_1 = 0.18347528

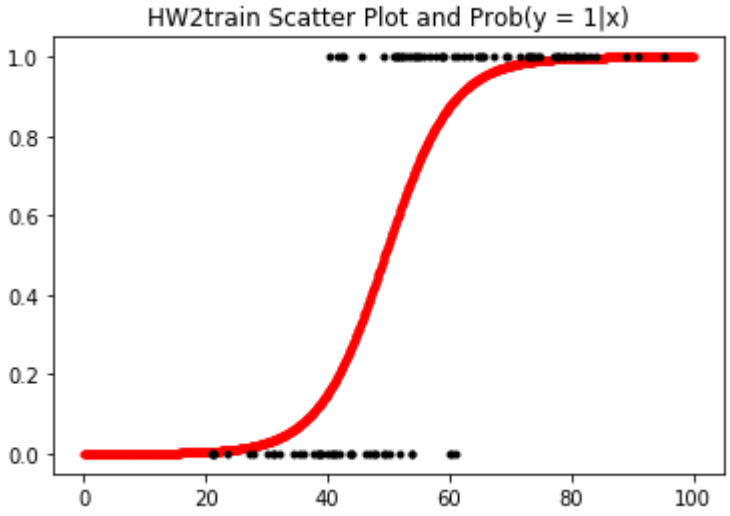
(2) Report the accuracy for HW2train (you can do this with the .score() function).

Accuracy is 0.86

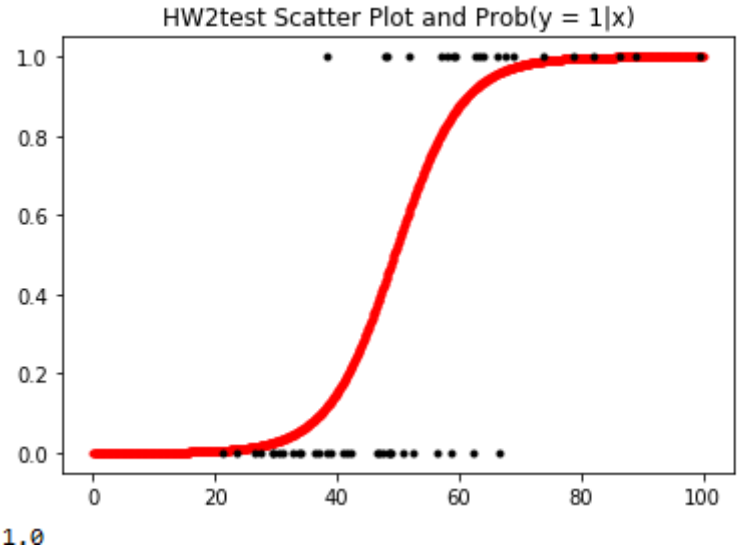
(3) Also, make a copy of the scatter-plot of the data-set HW2train plot. Add the function Prob(y = 1|x) on the plot.

- To plot the Prob(y = 1|x) function, you can first generate uniformly spaced values along the horizontal axis, such as with https://docs.scipy.org/doc/numpy/reference/generated/numpy.linspace.html 1000 values evenly spaced between 0 and 100 should be enough for a good picture.
- then determine Prob(y = 1|x) for each of those evenly spaced points. One way to obtain this is the .predict proba() function https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression.predict_proba using those 1000 evenly spaced values as inputs; that function will return an array with Prob(y = 0|x) for the first column and Prob(y = 1|x) for the second; use the second column. Alternatively, you can use the β_0 and β_1 coefficients to calculate Prob(y = 1|x) by yourself.

The scatter plot markers of the HW2train data should be plotted on top of the function Prob(y = 1|x) (i.e. in the foreground), so it is not painted over by the Prob(y = 1|x) function. Title this plot 'HW2train Scatter Plot and Prob(y = 1|x)'.



(4) Next make another scatter plot titled 'HW2test Scatter Plot and Prob($y = 1|x$)' just like the previous plot except where you plot the HW2test data instead of HW2train data. Include the same Prob($y = 1|x$) function as in the previous plot.




```
1.0
```

(5) Report the total accuracy for the HW2test data.

The accuracy is 0.82.

C. Now we will try *k*-nearest neighbors. Our predictions will be based on the data set HW2train and odd-values of *k*, using majority vote. Since we only have a single (one-dimensional) feature, *x*, we will measure the distance between sample *i* and a new sample using the absolute difference, $|x(i) - x(\text{new})|$.
You can implement *k*nn manually or by using built in functions. For Python, you can use https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClass html#sklearn.neighbors.KNeighborsClassifier Some usage notes

- the argument 'n neighbors' is *k*, so set 'n neighbors'=1 when you just want to use the nearest neighbor.
- set the argument 'weights'='uniform' (for this assignment we will just use uniform weights, but we encourage you to explore what happens with 'weights'='distance' which uses a built-in distance weighting or one you make yourself) • the argument 'algorithm' effects how many distances are computed to find the nearest neighbors for a new sample. Use 'auto' for this assignment.

(1) For each value of $k \in \{1,3,9\}$

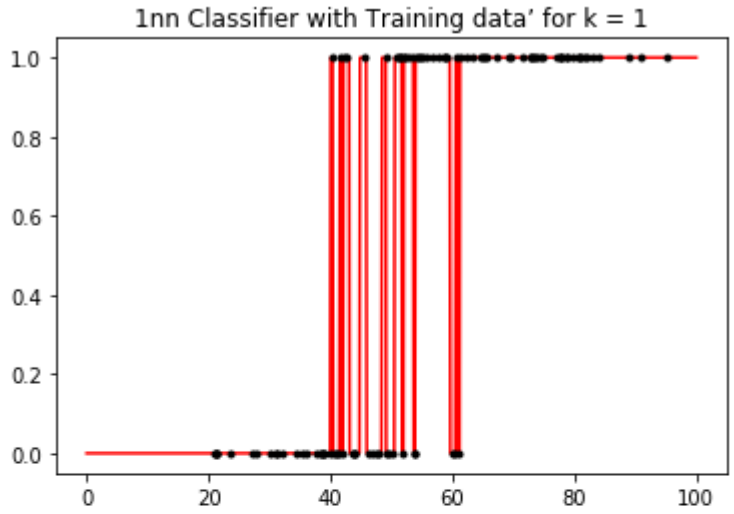
a. Fit the *k*nn classifier using the HW2train data set. Report the training accuracy (if using Python, you can use the .score()); briefly mention how you calculated it, such as if you used .score() or some other way. I used .score() to calculate the accuracy.

b. Make a plot of the classifier's prediction $y(x)$ function. This should be a b step-function (piece-wise constant), though your plot of the function can have steeply slanted lines instead of perfectly vertical jumps. Use 1000 linearly spaced values along the horizontal axis, like you did for the logistic curve in 5.B.(3).
Also plot HW2train data in the foreground (as a scatter plot). Use the title '1nn Classifier with Training data' for $k = 1$ and similar titles for other *k*.

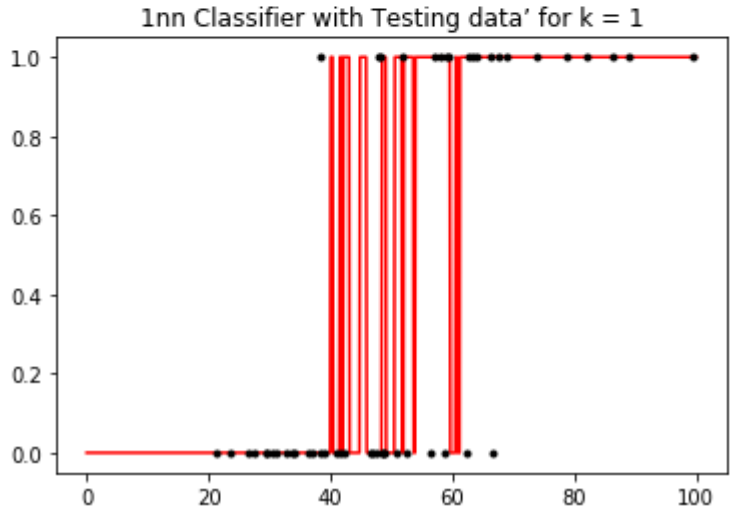
c. Report the total accuracy for HW2test data set.

d. Make another plot, also with the classifier's prediction $y(x)$ function, but show the HW2test data instead. Use the title '1nn Classifier with Testing data' for $k = 1$ and similar titles for other *k*.

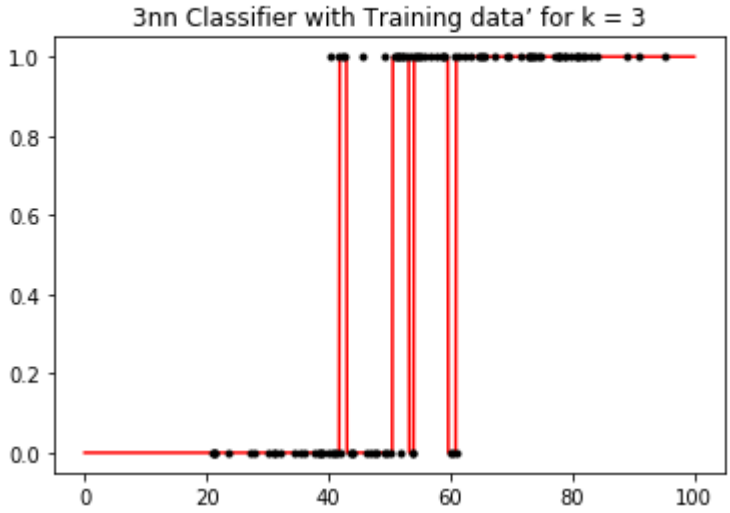
```
Training Accuracy 1.0
```



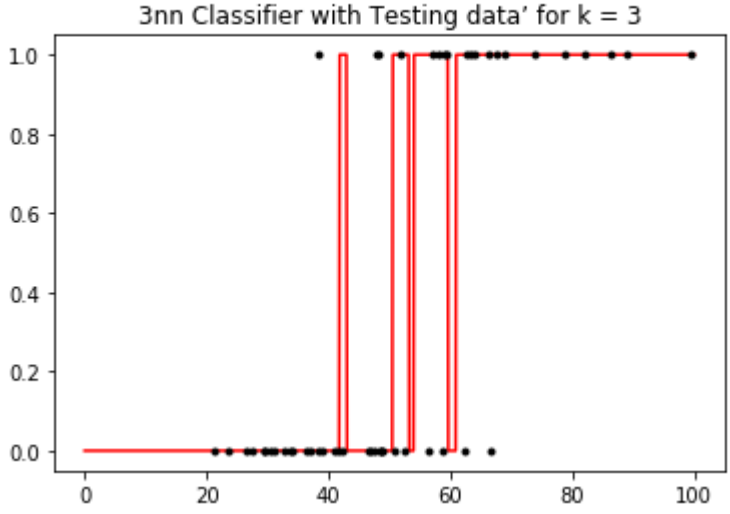

```
Testing Accuracy 0.7
```



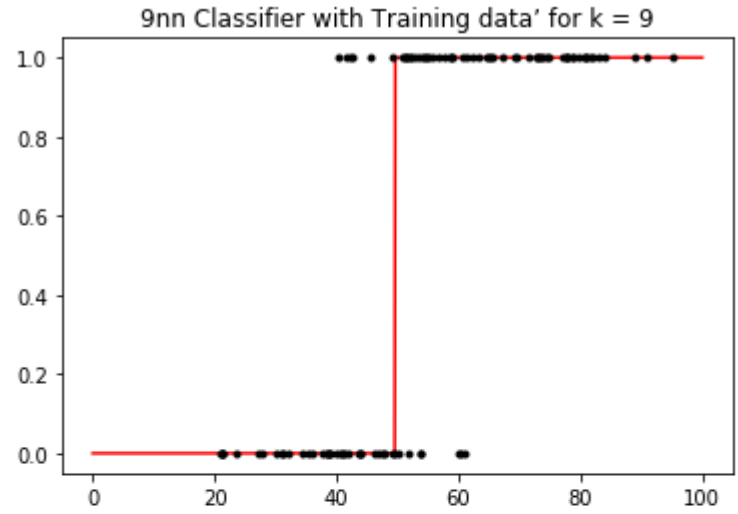
```
Training Accuracy 0.9
```



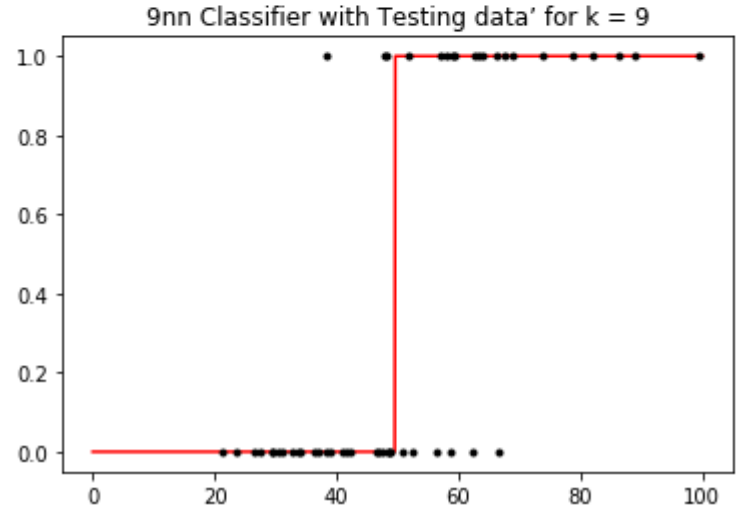

```
Testing Accuracy 0.8
```



```
Training Accuracy 0.86
```



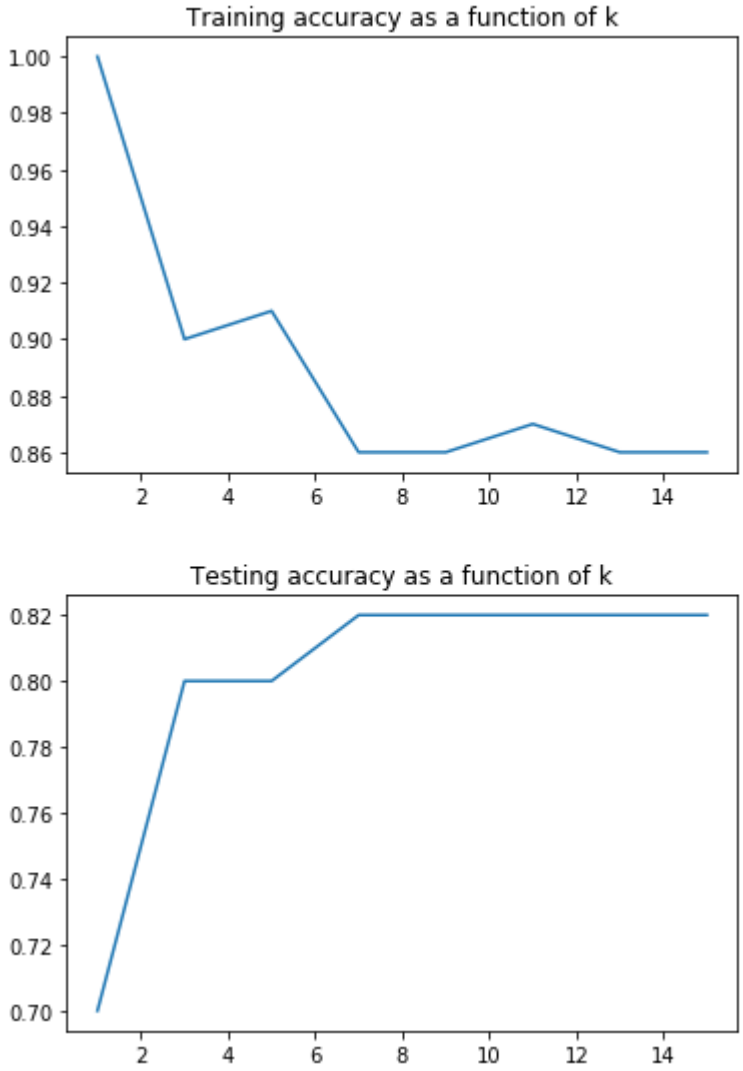

```
Testing Accuracy 0.82
```



(2) Make a plot with the title 'Training accuracy as a function of *k*' where the horizontal axis is the parameter *k* with the odd-numbered values {1,3,5,...,13,15}. The vertical axis should be the ~~training~~ accuracy for the HW2Train data set using the *k*nn classifier fit on the HW2Train data.

(3) Make a plot with the title 'Testing accuracy as a function of *k*' where the horizontal axis is the parameter *k* with the odd-numbered values {1,3,5,...,13,15}. The vertical

axis should be the ~~training~~ accuracy for the HW2Test data set using the *k*nn classifier fit on the HW2Train data.



D. In about 4-6 sentences, comment on the performance of the different nearest neighbor classifiers for the different *k* values you used, including whether you see any evidence of over-fitting or under-fitting, and how they compare to the logistic regression classifier, and any other note-worthy aspects.

When K=1, the training data is fit perfectly, however, this causes overfitting to occur and the test data is fit poorly. The testing data fit gets better as K increases to about 7, then the fit remains constant on the testing data. As K increases, the fit on the training data decreases.

When K-nearest neighbors is compared to logistic regression, they both achieve a testing accuracy of 0.82, however, K must be selected to be 7 or more. This implies that when choosing a K, one must select K carfully as to not overfit the data.