API 222 Problem Set 1

Machine Learning and Big Data Analytics: Fall 2024

Hun (Donghun) Kim

Due at 11:59am on September 25 - submit on Gradescope

This problem set is worth 30 points in total. To get full credit, submit your code along with a write-up of your answers. This should either be done in R Markdown or Jupyter Notebook, submitted in one knitted PDF.

Brief survey (0 pts)

Please fill out this brief (ungraded) survey to help Professor Saghafian and the teaching assistants get to know you. The link to the survey can also be found on Canvas under "Quizzes".

Conceptual Questions (15 pts)

1. For each of the following questions, state: (6 pts)

`(1) Whether it is a regression question or a classification question`

`(2) Whether we are interested in inference or prediction`

(a) A health organization is seeking to improve mental health services in a rural area. They want to identify individuals at high risk (binary) of developing stress-related disorders. They have demographic data and survey responses about lifestyle and stress levels.

Answer: This is a classification question since the "high risk" is being seen as a binary feature. Using the given features the model will classify whether the individual is in the "high risk" category or not. This is more of a prediction question because the organization is interested in identifying individuals at high risk of developing stress-related disorders based on their demographic data and survey responses.

(b) An environmental agency is aiming to reduce water pollution in a coastal region. They want to identify the factories most likely to exceed a given pollution limit. They have data on factory outputs, waste management practices, and historical pollution levels.

Answer: This is a regression question because there is the word "most likely" in the question, which implies that the agency is interested in predicting the probability of factories exceeding a given pollution limit. This is more of a prediction question because the agency is interested in identifying the factories most likely to exceed a given pollution limit based on their data on factory outputs, waste management practices, and historical pollution levels.

(c) A team of researchers is investigating the impact of dietary changes on body mass index (BMI) among middle-aged adults. They first implement a program promoting a balanced diet and then measure the change in the participants' BMI over six months.

Answer: Since the researchers are investigating the impact of dietary changes on BMI, this is a regression question. This is more of an inference question because the researchers are interested in understanding the relationship between dietary changes and BMI among middle-aged adults.

2. Flexible models versus inflexible models (5 pts)

(a) Flexible models will generally have lower bias than inflexible models. True or False?

Answer: True, flexible models will generally have lower bias than inflexible models because they are able to capture more complex patterns in the data. However, flexible models may have higher variance, which can lead to overfitting.

(b) For the same very large number of observations, inflexible models will likely perform better than flexible models when the number of features is small. True or False?

Answer: True, inflexible models will likely perform better than flexible models when the number of features is small because they are less likely to overfit the data.

(c) If the underlying data generating process is linear, a flexible model will generally perform worse than an inflexible one. True or False?

Answer: True, if the underlying data generating process is truly linear, an inflexible model will generally perform better than a flexible model because the flexible model may overfit the data by capturing unnecessary noise in the data.

(d) Non-parametric models impose stronger assumptions on the underlying data generating process than parametric models. True or False?

Answer: False, non-parametric models impose weaker assumptions on the underlying data generating process than parametric models because they do not assume a specific functional form for the relationship between the features and the target variable. On the other hand parametric models assume a specific functional form of the relationship between input variables and target variable.

(e) KNN and linear regression are both parametric models, as they both have decision rules. True or False?

Answer: False, KNN is a non-parametric model because it does not assume a specific functional form for the relationship between the features and the target variable. Linear regression is a parametric model because it assumes a linear relationship between the input variables and the target variable.

3. The bias-variance tradeoff (4 pts)

(a) What does bias refer to in the machine learning context?

Answer: Bias refers to the error that is introduced by trying to fit a complex real-world problem with a simple model. It is the difference between the predicted values and the true values in the data. High bias models are too simple and may not capture the underlying patterns in the data.

(b) What does variance refer to in the machine learning context?

Answer: Variance refers to the unreliability of a model's predictions due to sensitivity to the training data. It is the variability of the model's predictions for a given data point. High variance models are too complex and may capture noise in the data.

(c) Now briefly describe the bias-variance tradeoff.

Answer: In general, when we use more flexible methods for machine learning algorithms, since it is able to capture more complex patterns in the data, the variance of the model increases and the bias decreases. On the other hand, when we use less flexible methods, the bias of the model increases and the variance decreases. The bias-variance trade-off is the balance between the bias and variance of a model. The goal is to find the optimal balance that minimizes both bias and variance to achieve the best predictive performance.

(d) Briefly explain the issue of overfitting in light of the bias-variance trade-off.

Answer: Overfitting occurs when a model learns the noise in the training data rather than the underlying patterns. This leads to a model that performs well on the training data but poorly on new, unseen data. Overfitting is often a result of a model that is too complex, leading to high variance and low bias. In the context of the bias-variance trade-off, overfitting occurs when the model has low bias and high variance, meaning it captures the noise in the training data rather than the underlying patterns.

Data Questions (15 pts)

Maternal health is a vital component of public health, as the well-being of mothers directly impacts the health of future generations. Early identification of risks during pregnancy can significantly improve outcomes by enabling timely medical interventions. The dataset provided includes health indicators for pregnant women—such as age, systolic and diastolic blood pressure, blood sugar levels, body temperature, and heart rate—along with their associated risk levels categorized as low, mid, or high risk. Developing a predictive model using this data is crucial because it allows healthcare providers to assess the likelihood of complications during pregnancy.

For any non-integer numbers, please report your numbers to exactly two decimal places for full credit.

1. Preliminary data exploration

(a) How many observations and variables are in the dataset (1 pt)?

```r
# your code here

Health <- read.csv("Maternal Health Risk Data Set.csv")
health_data <- Health

# get observation count
observation_cnt <- nrow(health_data)
variables_cnt <- ncol(health_data)

print(paste("Observation count: ", observation_cnt))
```

```
## [1] "Observation count:  1014"
```

```
print(paste("Variables count: ", variables_cnt))
```

```
## [1] "Variables count:  7"
```

(b) Are any of the columns categorical? If so, which ones (2 pt)?

```
# your code here

health_data$RiskLevel <- as.factor(Health$RiskLevel)
print(levels(health_data$RiskLevel))
```

```
## [1] "high risk" "low risk"  "mid risk"
```

```
# check for categorical columns
categorical_columns <- sapply(health_data, is.factor)
categorical_columns <- names(categorical_columns[categorical_columns == TRUE])

print(paste("Categorical columns: ", categorical_columns))
```

```
## [1] "Categorical columns:  RiskLevel"
```

(c) Compute the mean and standard deviation of `Age` for each level of `RiskLevel`. What do you notice (2 pt)?

```
# your code here
high_risk_data = health_data[health_data$RiskLevel == "high risk",]
mid_risk_data = health_data[health_data$RiskLevel == "mid risk",]
low_risk_data = health_data[health_data$RiskLevel == "low risk",]

mean_age_low_risk <- round(mean(low_risk_data$Age), 2)
sd_age_low_risk <- round(sd(low_risk_data$Age), 2)

mean_age_mid_risk <- round(mean(mid_risk_data$Age), 2)
sd_age_mid_risk <- round(sd(mid_risk_data$Age), 2)

mean_age_high_risk <- round(mean(high_risk_data$Age), 2)
sd_age_high_risk <- round(sd(high_risk_data$Age), 2)

print(paste("Mean Age for Low Risk: ", mean_age_low_risk, "Standard Deviation Age for Low Risk: ", sd_ag
```

```
## [1] "Mean Age for Low Risk:  26.87 Standard Deviation Age for Low Risk:  13.12"
```

```
print(paste("Mean Age for Mid Risk: ", mean_age_mid_risk, "Standard Deviation Age for Mid Risk: ", sd_ag
```

```
## [1] "Mean Age for Mid Risk:  28.36 Standard Deviation Age for Mid Risk:  12.55"
```

```
print(paste("Mean Age for High Risk: ", mean_age_high_risk, "Standard Deviation Age for High Risk: ", s
```

```
## [1] "Mean Age for High Risk:  36.22 Standard Deviation Age for High Risk:  13.03"
```

> Answer: The mean age for high-risk pregnancies is higher than the mean age for low-risk and mid-risk pregnancies. The standard deviation are similar across the three risk levels, indicating that the age distribution is relatively consistent within each risk level.

For the next few questions, set the seed to 222 and randomly put 20% of your observations in a test set and the remaining observations in a training set.

```
# your code here
set.seed(222)

# split data into training and test sets
test_indices <- sample(1:nrow(health_data), round(0.2 * nrow(health_data)))
train_indices <- which(!(1:(nrow(health_data)) %in% test_indices))

test_data <- health_data[test_indices, ]
train_data <- health_data[train_indices, ]
```

2. When you use your training data to build a KNN classification model (with k=5) that predicts `RiskLevel` using all other features available in the data, what is your total error rate? (2 pt)

```
# your code here
library(class)


# we must drop the RiskLevel column from the test data and train data
knn_model5 = knn(train = train_data[, -which(names(train_data) %in% c("RiskLevel"))],
                 test = test_data[, -which(names(test_data) %in% c("RiskLevel"))],
                 cl = train_data$RiskLevel,
                 k = 5)

error_rate <- mean(knn_model5 != test_data$RiskLevel)
print(paste("Total error rate: ", round(error_rate, 2)))
```

```
## [1] "Total error rate:  0.3"
```

3. Now use your training data to build a KNN model that classifies `RiskLevel` using only three variables: Age, SystolicBP, and DiastolicBP.

(a) What is your total error rate (1 pt)?

```
# your code here
knn_model_limited = knn(train = train_data[, c("Age", "SystolicBP", "DiastolicBP")],
                        test = test_data[, c("Age", "SystolicBP", "DiastolicBP")],
                        cl = train_data$RiskLevel,
                        k = 5)

error_rate_limited <- mean(knn_model_limited != test_data$RiskLevel)
print(paste("Total error rate with limited features: ", round(error_rate_limited, 2)))
```

```
## [1] "Total error rate with limited features:  0.33"
```

(b) Check classification accuracy by class. What do you notice? Is your error rate being driven primarily by one class? (2 pt)

```r
# your code here
low_risk_indices <- which(test_data$RiskLevel == "low risk")
mid_risk_indices <- which(test_data$RiskLevel == "mid risk")
high_risk_indices <- which(test_data$RiskLevel == "high risk")

error_rate_low_risk <- mean(knn_model_limited[low_risk_indices] != test_data[low_risk_indices,]$RiskLeve
error_rate_mid_risk <- mean(knn_model_limited[mid_risk_indices] != test_data[mid_risk_indices,]$RiskLeve
error_rate_high_risk <- mean(knn_model_limited[high_risk_indices] != test_data[high_risk_indices,]$Risk

print(paste("Error rate for low risk: ", round(error_rate_low_risk, 2)))
```

```
## [1] "Error rate for low risk:  0.23"
```

```r
print(paste("Error rate for mid risk: ", round(error_rate_mid_risk, 2)))
```

```
## [1] "Error rate for mid risk:  0.44"
```

```r
print(paste("Error rate for high risk: ", round(error_rate_high_risk, 2)))
```

```
## [1] "Error rate for high risk:  0.33"
```

Answer: The error rate is highest in “mid risk” class. This suggests that the model may have more difficulty distinguishing between “mid risk” and the other classes.

(c) Provide some intuition for what it means that your KNN model with 3 features was “more predictive” than your KNN model with all of the features. (1 pt)

Answer: The KNN model with 3 features can be said to be more predictive than the KNN model with all features when it is able to get better accuracy despite having less features. By using only the most important features, the model can be able to reduce the noise in the data and make more accurate predictions.

4. Complete the code below to search for an optimal k. What does this tell you about the optimal flexibility of the decision boundary (2 pt)

```r
error_rates <- c()

for (k in 1:20) {

  # train knn model
  knn_model <- knn(train = train_data[, c("Age", "SystolicBP", "DiastolicBP")],
                   test = test_data[, c("Age", "SystolicBP", "DiastolicBP")],
                   cl = train_data$RiskLevel,
                   k = k)
  # save error rate for this
  error_rates[k] <- mean(knn_model != test_data$RiskLevel)
}

# find the optimal k
optimal_k <- which.min(error_rates[2:20])
optimal_k
```

```
## [1] 3
```

> Answer: The optimal k turned out to be 3. We excluded the 1 since it will automatically give the minimal error rate due to its high flexibility.

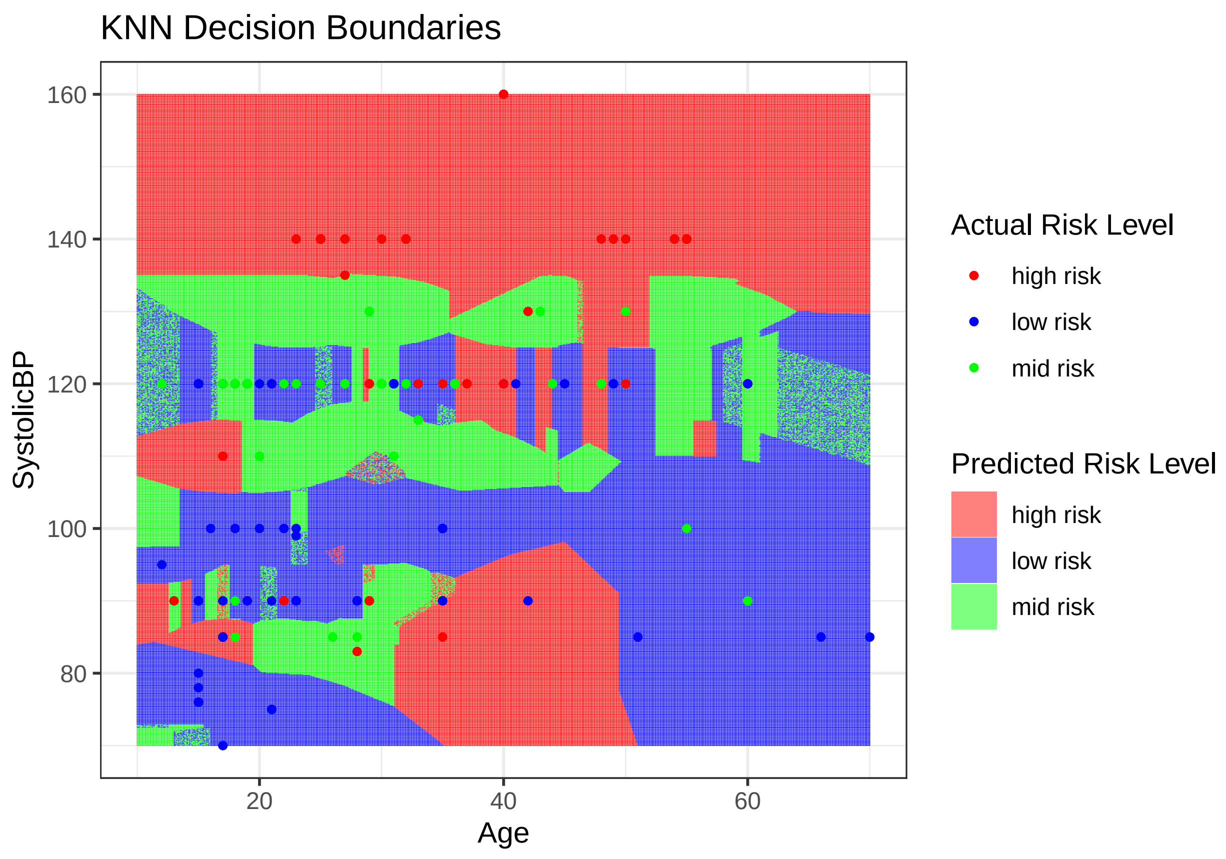
5. It turns out that we are actually only interested in predicting mothers who are "high risk". Create a new binary variable called `HighRisk` which takes the value 1 when `data$RiskLevel == "high risk` and takes the value 0 when risk is not "high risk". Train a KNN model that uses only `Age`, `SystolicBP`, and `DiastolicBP` to predict this new binary variable. What is your misclassification error rate? (2 pts)

```
# your code here

health_data$HighRisk <- ifelse(health_data$RiskLevel == "high risk", 1, 0)

high_risk_train <- sample(1:nrow(health_data), 0.8 * nrow(health_data))
train_data_high_risk <- health_data[high_risk_train, ]
test_data_high_risk <- health_data[-high_risk_train, ]

knn_model_high_risk <- knn(train = train_data_high_risk[, c("Age", "SystolicBP", "DiastolicBP")],
                           test = test_data_high_risk[, c("Age", "SystolicBP", "DiastolicBP")],
                           cl = train_data_high_risk$HighRisk,
                           k = 5)

error_rate_high_risk <- mean(knn_model_high_risk != test_data_high_risk$HighRisk)
print(paste("Misclassification error rate for high risk: ", round(error_rate_high_risk, 2)))
```

```
## [1] "Misclassification error rate for high risk:  0.17"
```

Challenge exercise (just for fun only– completely optional): Use the code from this blog post in the Visualizing Decision Boundaries section to create a plot that shows your prec the decision boundaries in a model that only uses k=3 and only 2 features: `Age`, `SystolicBP`.

```
# your code here

library(ggplot2)

x_range <- seq(from = min(health_data$Age), to = max(health_data$Age), by = 0.1)
y_range <- seq(from = min(health_data$SystolicBP), to = max(health_data$SystolicBP), by = 0.1)

grid <- expand.grid(Age = x_range, SystolicBP = y_range)

grid$RiskLevel <- knn(train = train_data[, c("Age", "SystolicBP")],
                      test = grid,
                      cl = train_data$RiskLevel,
                      k = 3)

ggplot() +
  geom_tile(data = grid, aes(x = Age, y = SystolicBP, fill = RiskLevel), alpha = 0.5) +
  geom_point(data = test_data, aes(x = Age, y = SystolicBP, color = RiskLevel), size = 1) +
  scale_fill_manual(values = c("low risk" = "blue", "mid risk" = "green", "high risk" = "red")) +
  scale_color_manual(values = c("low risk" = "blue", "mid risk" = "green", "high risk" = "red")) +
  labs(
    title = "KNN Decision Boundaries",
    x = "Age",
    y = "SystolicBP",
    fill = "Predicted Risk Level",
    color = "Actual Risk Level"
  ) +
  theme_bw()
```


KNN Decision Boundaries
SystolicBP
Age
Actual Risk Level
high risk
low risk
mid risk
Predicted Risk Level
high risk
low risk
mid risk