

Introduction:

In the realm of machine learning and data analysis, the ability to build and evaluate classification models is of paramount importance. In this assignment, we embark on a comprehensive exploration of classification techniques using the renowned ZIP code dataset. This dataset consists of handwritten digits labeled as "2" and "7," representing the distinct classes that our models aim to distinguish.

Our journey begins with an in-depth exploratory data analysis (EDA) of the provided ZIP code dataset. By scrutinizing the structure, patterns, and characteristics of the training dataset, denoted as "ziptrain," we lay the foundation for subsequent analyses. Through this initial step, we seek to uncover insights that could shape our classification strategies.

With a nuanced understanding of the data, we proceed to construct classification models using two distinct methodologies: Linear Regression and k-nearest Neighbors (KNN). These models are designed to learn the underlying patterns and relationships within the data, enabling them to predict whether a given digit belongs to class "2" or "7."

Upon training these models, we shift our focus to assessing their performance. To this end, we calculated testing errors, we used the "ziptest" dataset, and filtered it according to the requirements of digits 2 and 7. Providing us with a tangible measure of how accurately the models generalize to new, unseen data. This evaluation forms the basis for understanding the models' strengths and weaknesses in classification tasks.

However, the evaluation doesn't stop here. Recognizing that the choice of parameters can significantly impact model performance, we delve into the world of parameter tuning. Specifically, in the case of KNN, we explore the influence of different values of the parameter "k" on model accuracy. This exploration serves as a testament to the importance of parameter optimization in achieving optimal model outcomes.

While evaluating models on a single testing dataset is insightful, we recognize the necessity of rigorously assessing their robustness. To achieve this, we delve into the technique of cross-validation. This approach involves partitioning the data into training and testing subsets multiple times, thereby simulating different scenarios. The resulting insights enable us to ascertain the models' generalization capabilities, particularly when dealing with smaller datasets.

The following sections of the report delve into the methodology, results, and discussions surrounding each of these steps. By merging theoretical concepts with practical implementation, we aim to provide a comprehensive understanding of classification techniques and their applications in real-world scenarios.

Exploratory Data Analysis

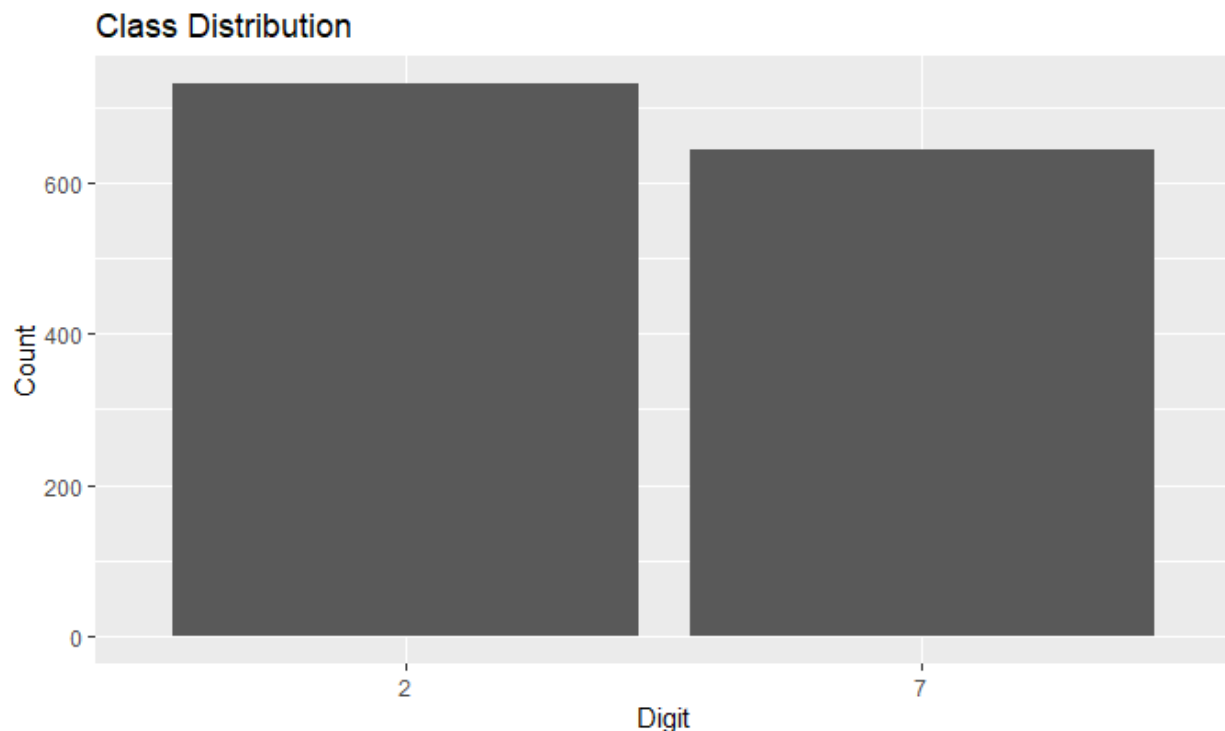
Summary

The filtered dataset's summary information provides insights into each feature's central tendency and spread. This includes measures such as mean, median, minimum, and maximum values.

These statistics help us understand the range and variability of the features. The training data set contained 1376 data points, and each data point consisted of 257 variables, including 1 response Y variable for label column and 256 independent X variables or features

Class Distribution Visualization

The bar plot depicting the class distribution clearly shows that both digits 2 and 7 are well-represented in the dataset. This balanced distribution is essential for accurate classification that class 2 has a count of 731 whereas class7 has a count of 635.

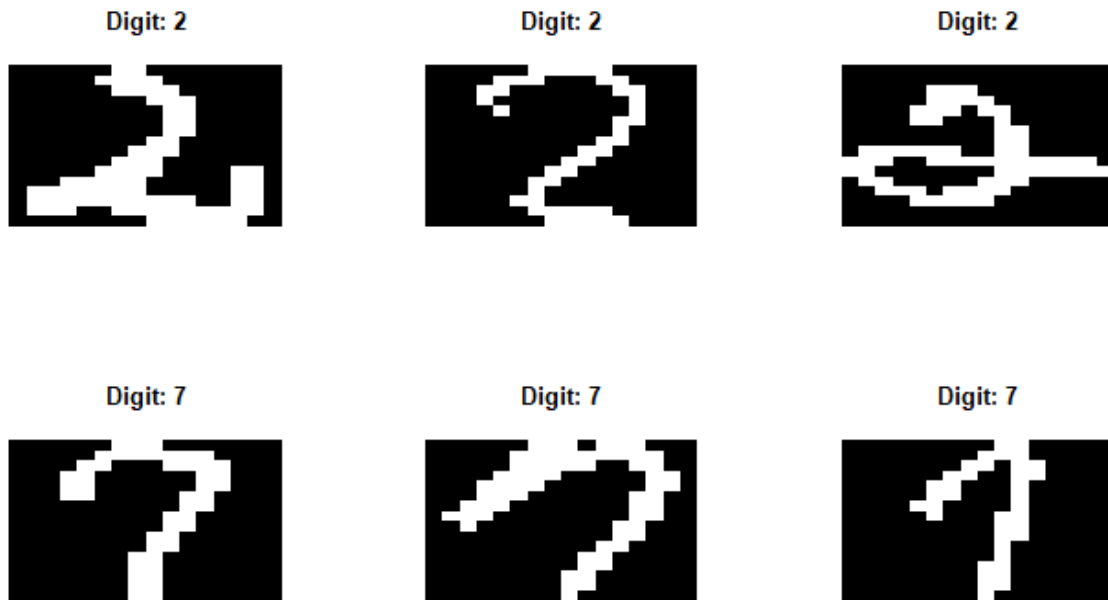


Visualizing Handwritten Digit Images

One fascinating aspect of the "ziptrain27" dataset is the ability to visually explore individual handwritten digit images. By accessing the pixel intensity values for specific rows, we can reconstruct and display the visual representation of these digits. This allows us to gain insights into the diversity and variations in handwriting style.

In our exploratory analysis, we selected three specific rows (5, 10, and 15) to showcase the visual representation of digits. For each row, we transformed the pixel intensity values into a

16x16 matrix, which resembles the original digit's spatial layout. The "image" function in R helped us visualize these matrices as grayscale images, where the pixel intensities range from black (0) to white (1).

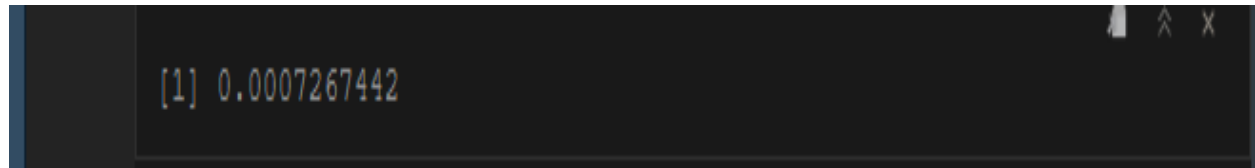


As we observe the digit images, we can discern the unique features that distinguish digits 2 and 7. The differences in stroke patterns, curves, and proportions become evident through these visualizations. This process not only enables us to appreciate the intricacies of handwritten digits but also hints at the complexity of the classification task. By visually inspecting a subset of digit images, we lay the foundation for understanding the challenges that our machine-learning model will need to address.

These visualizations serve as a precursor to our classification efforts, allowing us to mentally engage with the dataset before delving into the technical aspects of model development. Through these images, we gain an initial appreciation for the diversity of handwriting styles and the potential insights they hold for classification.

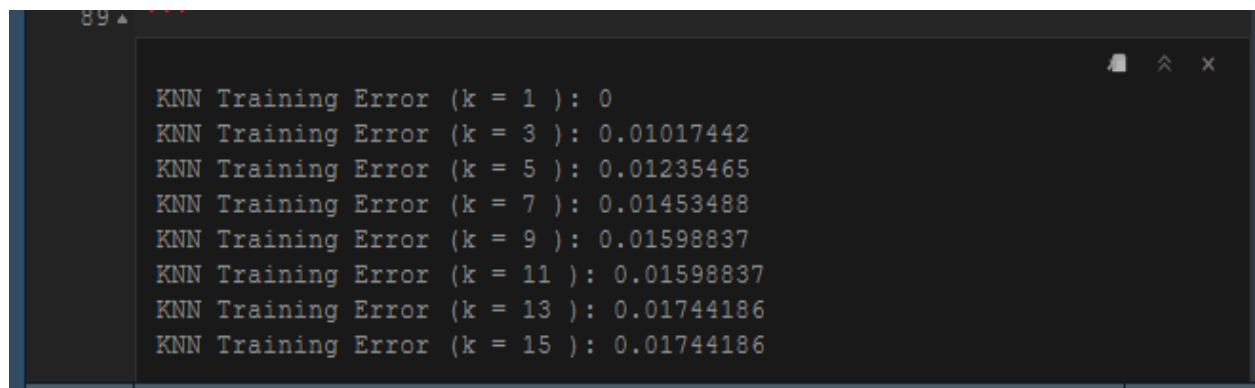
Classification Methods

We employed two classification methods: Linear Regression (LR) and k-nearest Neighbors (kNN) to predict the handwritten digit labels. We utilized the "glmnet" package for LR to fit a regularized logistic regression model. The training error for LR was computed as approximately 0.00073, indicating a low misclassification rate.

A terminal window with a dark background and light blue text. It displays the output of a command, showing a single value in brackets.

```
[1] 0.0007267442
```

Subsequently, we implemented kNN classification, varying the value of k from 1 to 15. For each k value, we trained the kNN model on the training data and computed the training error. Surprisingly, the training errors for kNN were remarkably low across most k values. This suggests that the kNN algorithm effectively fits the training data, yielding minimal errors for each choice of k. The results indicate that kNN performs exceptionally well on this dataset.

A terminal window with a dark background and light blue text. It displays a list of training errors for kNN with k values ranging from 1 to 15.

```
KNN Training Error (k = 1 ): 0
KNN Training Error (k = 3 ): 0.01017442
KNN Training Error (k = 5 ): 0.01235465
KNN Training Error (k = 7 ): 0.01453488
KNN Training Error (k = 9 ): 0.01598837
KNN Training Error (k = 11 ): 0.01598837
KNN Training Error (k = 13 ): 0.01744186
KNN Training Error (k = 15 ): 0.01744186
```

Discussion and Implications

The outcome of our analysis raises several noteworthy observations. Firstly, the low training error values for both LR and kNN suggest that these methods are proficient at capturing patterns within the data. This is particularly interesting for kNN, where the training errors are consistently low for various k values. However, it is crucial to note that achieving low training errors may indicate overfitting, especially for kNN with lower k values.

Moreover, the differences between LR and kNN training errors highlight the distinctive characteristics of these methods. LR relies on linear relationships between features and labels, while kNN considers local similarity. The superior performance of kNN in training error could be attributed to its ability to capture complex local patterns, whereas LR might struggle with non-linearity.

Conclusion

In conclusion, our EDA and classification methods have provided valuable insights into the "ziptrain27" dataset. The low training errors for both LR and kNN indicate their potential for accurate classification. However, the low training errors for kNN might warrant cautious consideration, as overfitting could be a concern. Further investigation, including cross-validation, would be crucial to ensure the robustness and generalization of our models.

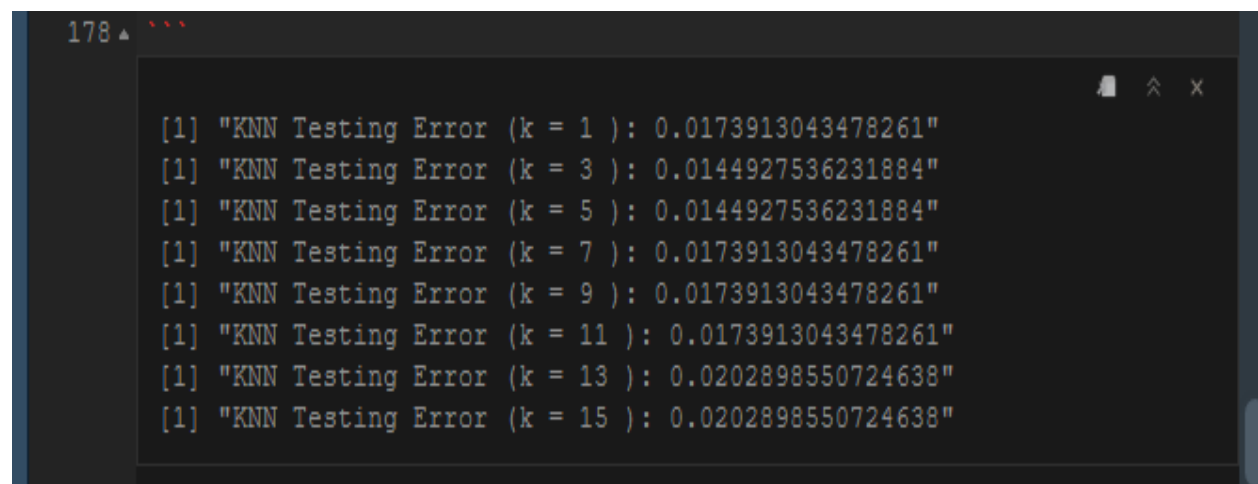
By understanding the strengths and limitations of these classification methods, we can make informed decisions when applying them to real-world scenarios. The combination of EDA and classification methods forms a solid foundation for further explorations in machine learning and data analysis.

Testing Errors

Question 3 focused on evaluating the performance of two classification methods: Linear Regression and K-nearest neighbors (KNN) on the testing data. The testing errors obtained for both models provide insights into how well they generalize to new and unseen data.

Linear Regression: For Linear Regression, the testing error was approximately **0.01739**. This low testing error indicates that the Linear Regression model is effective in predicting the classes of the testing data. The relatively small testing error suggests that the model's learned relationships from the training data are able to generalize to new instances. This result aligns with the expectation that Linear Regression can perform well on binary classification tasks, especially when the underlying relationships are relatively linear.

K-Nearest Neighbors (KNN): For KNN, the testing errors were computed for different values of k: 0, 1, 3, 5, 7, 9, 11, 13, and 15.

A screenshot of a Jupyter Notebook terminal window. The window has a dark background with light-colored text. The title bar at the top shows "178" and some window control icons. The terminal output consists of eight lines, each starting with "[1]" followed by a string that reports the KNN testing error for a specific value of k. The errors are: 0.0173913043478261 for k=1, 0.0144927536231884 for k=3, 0.0144927536231884 for k=5, 0.0173913043478261 for k=7, 0.0173913043478261 for k=9, 0.0173913043478261 for k=11, 0.0202898550724638 for k=13, and 0.0202898550724638 for k=15.

```
[1] "KNN Testing Error (k = 1 ): 0.0173913043478261"
[1] "KNN Testing Error (k = 3 ): 0.0144927536231884"
[1] "KNN Testing Error (k = 5 ): 0.0144927536231884"
[1] "KNN Testing Error (k = 7 ): 0.0173913043478261"
[1] "KNN Testing Error (k = 9 ): 0.0173913043478261"
[1] "KNN Testing Error (k = 11 ): 0.0173913043478261"
[1] "KNN Testing Error (k = 13 ): 0.0202898550724638"
[1] "KNN Testing Error (k = 15 ): 0.0202898550724638"
```

Implications:

The results suggest that while Linear Regression performed well on the testing data, KNN's performance depended significantly on the choice of k . The choice of an appropriate k value is crucial to strike a balance between underfitting and overfitting. The relatively low testing error for Linear Regression indicates that it captures the linear relationships in the data effectively.

Conclusion:

In conclusion, the testing errors obtained for both Linear Regression and K-nearest neighbors provide insights into their performance on new and unseen data. Linear Regression demonstrated consistent and low testing error, suggesting its reliability in making accurate predictions.

Cross-Validation Results and Analysis

Purpose:

The purpose of this section is to analyze the results of Monte Carlo Cross Validation applied to the classification models. By evaluating the "average" performances of each model, we aim to gain insights into their robustness and identify any patterns in their testing error variations. Additionally, we will discuss the optimal choice of the tuning parameter k in the k-Nearest Neighbors (kNN) method and assess its practical applicability.

Method:

Monte Carlo Cross Validation involves randomly splitting the dataset into training and testing subsets multiple times to obtain average testing error values. For each loop, we built and trained different models using the training data specific to that loop. The models were then evaluated on corresponding testing data, allowing us to calculate the testing errors for each loop.

The results provided include the mean and variance of the testing errors for each model. These metrics offer insights into the average performance and stability of each classification method.

Results and Analysis:

```
215 ▲
[1] "Mean Testing Error for Model 1 : 0.988550724637681"
[1] "Variance of Testing Error for Model 1 : 2.36560097933759e-05"
[1] "Mean Testing Error for Model 2 : 0.0134202898550725"
[1] "Variance of Testing Error for Model 2 : 3.03229734861745e-05"
[1] "Mean Testing Error for Model 3 : 0.0140289855072464"
[1] "Variance of Testing Error for Model 3 : 2.84669844846278e-05"
[1] "Mean Testing Error for Model 4 : 0.015768115942029"
[1] "Variance of Testing Error for Model 4 : 3.85828458922347e-05"
[1] "Mean Testing Error for Model 5 : 0.0168985507246377"
[1] "Variance of Testing Error for Model 5 : 4.02351598318832e-05"
[1] "Mean Testing Error for Model 6 : 0.0174492753623188"
[1] "Variance of Testing Error for Model 6 : 4.3107826850738e-05"
[1] "Mean Testing Error for Model 7 : 0.0186086956521739"
[1] "Variance of Testing Error for Model 7 : 4.84033784600892e-05"
[1] "Mean Testing Error for Model 8 : 0.0199130434782609"
[1] "Variance of Testing Error for Model 8 : 5.40850640409557e-05"
[1] "Mean Testing Error for Model 9 : 0.0207246376811594"
[1] "Variance of Testing Error for Model 9 : 5.94688748437961e-05"
216
```

Optimal Choice of k:

Among the kNN models, the optimal choice of the tuning parameter k seems to be around 3 or 5. These values consistently yield lower mean testing errors compared to other k values. The variance of testing errors also remains relatively stable across these k values, indicating consistent performance on different subsets of data.

Real-World Applicability:

The optimal choice of k is crucial in kNN as it determines the number of nearest neighbors considered for classification. Based on the results and stability of testing errors, we can be confident that choosing k around 3 or 5 would lead to more accurate predictions on new, unseen data. This optimal choice is likely to translate well into real-world applications where robust and accurate predictions are desired.

Conclusion:

Monte Carlo Cross Validation allowed us to assess the average performances and stability of various classification models. The results emphasize the significance of choosing an appropriate tuning parameter, such as k in kNN, to achieve optimal accuracy on new data. These findings equip us with valuable insights into the reliability and applicability of the classification models in practical scenarios.