DIT406: Assignment 2: Regression and Classification

Calvin Smith, Emir Zivcic

Chalmers University of Technology/University of Gothenburg

November 23, 2021

1 Regression - Landvetter selling prices

a)

Find a linear regression model that relates the living area to the selling price. If you did any data cleaning step(s), describe and explain why you did that The only columns of interest were living area and selling price which had no missing values, thus no cleaning had to be done. Otherwise **dropna()** could've been used if rows with missing values existed.

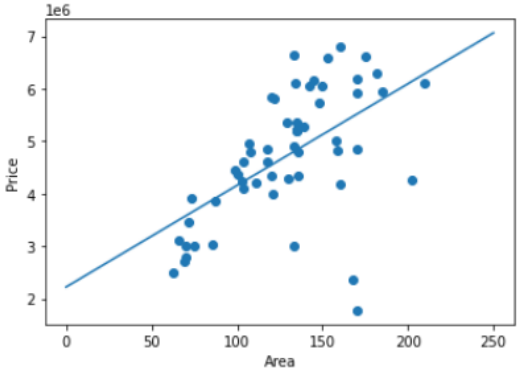

Figure 1: Linear regression model

b)

What are the values of the slope and intercept of the regression line? The slope is the coefficient of the linear regression model and has the value 19400, rounded up to hundreds. The intercept is essentially the base price (Area = 0) and has the value 2.22 million.

c)

Use this model to predict the selling prices of houses which have living area 100 m² , 150 m² and 200 m² The model predicts the price of a 100m² house to sell for 4.16 million, a 150m² house to sell for 5.13 million and a house of 200m² to sell for 6.10 million.

d)

Draw a residual plot Residuals were calculated by subtracting the predicted value for a house with a certain size with its actual selling price. A negative residual means it was sold for less than predicted and vice versa.

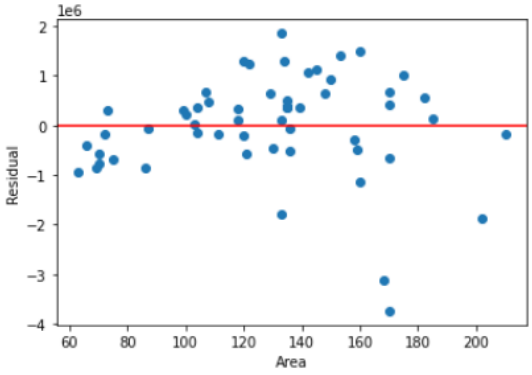

Figure 2: Residual plot

e)

Discuss the results, and how the model could be improved. There are obvious issues with the high variance residuals that can be seen in the residual plot. This could be due to outliers that exist in the data set. A possible improvement would be to remove these outliers from the data set since they do not follow the trend of the population.
The model was created from two variables and disregard other variables such as the age of the estate and the size of the land. A newly built house will be more valuable than the same house if it was 60 years old or more valuable if the land size is huge rather than small. We can see when creating a model over the variables: living area, rooms, age, and land size that the residual sum goes down by around 17% and the absolute sum goes down by around 24%. This result shows that we only had outliers since we ignored important variables in our first model.

2 Classification - Iris data set

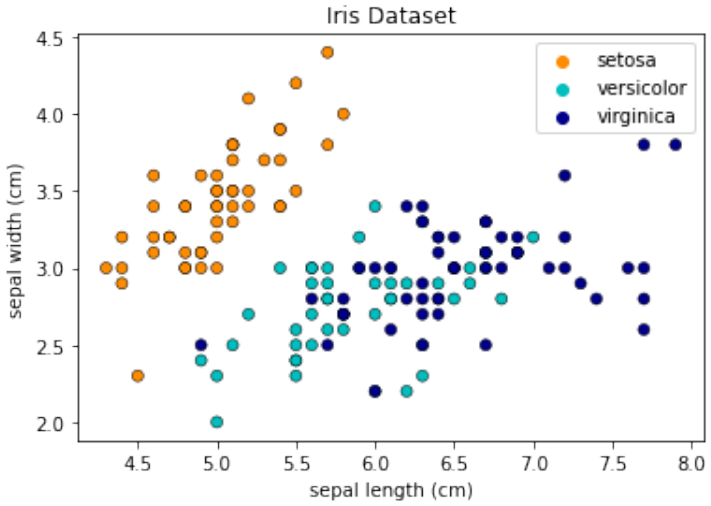

Figure 3: Overview of the iris dataset

This task consist of evaluating two different methods, logistic regression and KNN, for classifying the iris dataset. An overview of the data can be seen in figure 3. In preparation of the classification task we have split the data into a training and testing set with a split ration of 80:20. Also, we have chosen two use only two features when training our models, **sepal length(cm)** and **sepal width(cm)**. This approach gives us models with less accuracy (although not a huge difference), but is better for illustrative purposes.

a)

Use a confusion matrix to evaluate the use of logistic regression to classify the iris data set.

The predictions from our logistic regression model on the testing data can be seen in the confusion matrix from figure 4. We achieved and accuracy of 0.87. Furthermore, figure 5 shows the decision boundaries produced by our model together with a scatterplot of the actual labels. We notice that the model manages to separate **setosa** from the two other classes, while it has a more difficult time of separating **versicolor** and **virginica** which have some overlap.

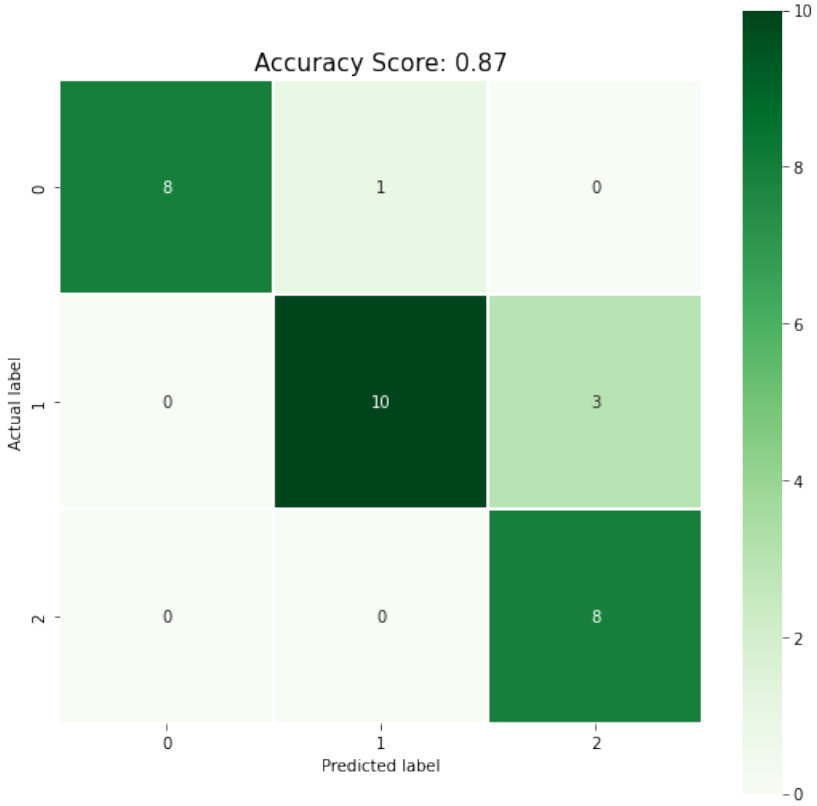

Figure 4: Confusion matrix for logistic regression.

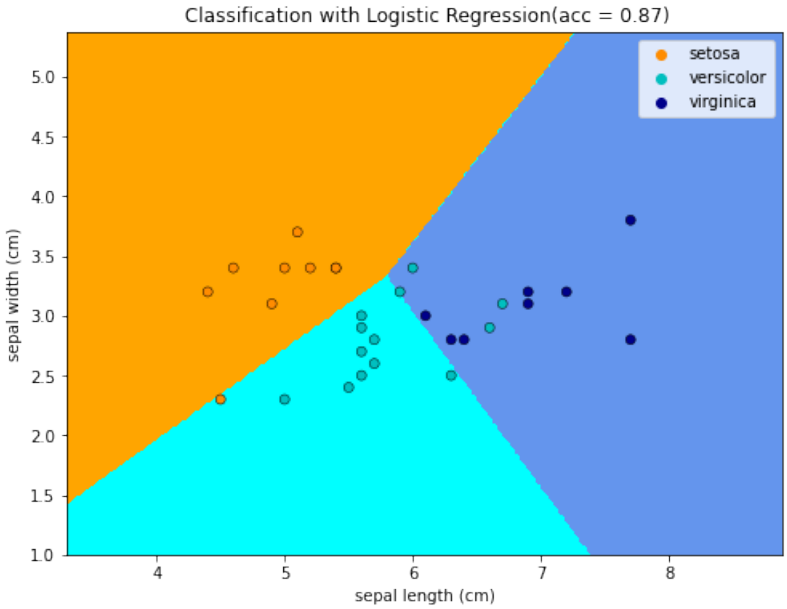

Figure 5: Results from classifying the iris dataset using logistic regression.

b)

Use k-nearest neighbours to classify the iris data set with some different values for k, and with uniform and distance-based weights. What will happen when k grows larger for the different cases? Why?

Figure 6 shows how the accuracy changes with the number of neighbors (k) in our KNN-models. KNN with uniform based weights peaks at $k = 25$ with an accuracy of 0.93 and then slowly declines. KNN with distance based weights peaks at $k = 8$ with an accuracy of 0.87 until it stabilizes and eventually reaches a steady state.

So why does the accuracy decrease/stabilize as we increase k for the different cases? The idea behind KNN is that observations that lie close together are similar, so we classify points based on the classes of its neighboring points. So the main question is: how many neighboring points does it make sense to include for my classification problem? The uniform based weights method means that a point gets classified according to the majority vote among its k-nearest neighbors, with all points weighted equally. When we increase the number of neighbors we include points that are further away from our observation and will be considered in the majority vote. If we choose a k that is too large, we risk to include points that are very different from our observation and thus our prediction accuracy decreases. This is quite clearly illustrated in figure 6 where

we see that the accuracy quickly rises, reaches a peak and then starts to decline. The distance based method places weights on each point in the neighborhood so that points in the neighborhood that are closer to our observation will be weighted higher than points further away. In this case, we see from figure 6 that the accuracy increases, reaches a peak (at $k = 8$) and then more or less stabilizes. This is because, as we increase k from the peak, the neighborhood area will expand and include more points, these points will obviously be further away from the observation but their weights will be so small that the classification wont be affected, thus we reach a steady state.

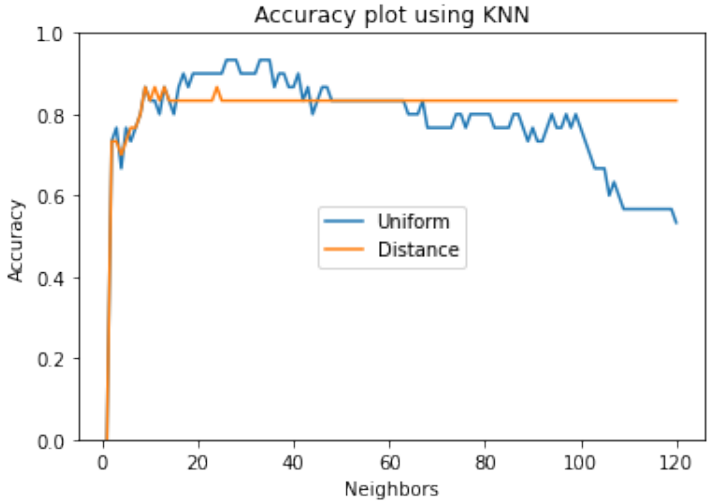

Figure 6: Accuracy for different values of k in KNN classification.

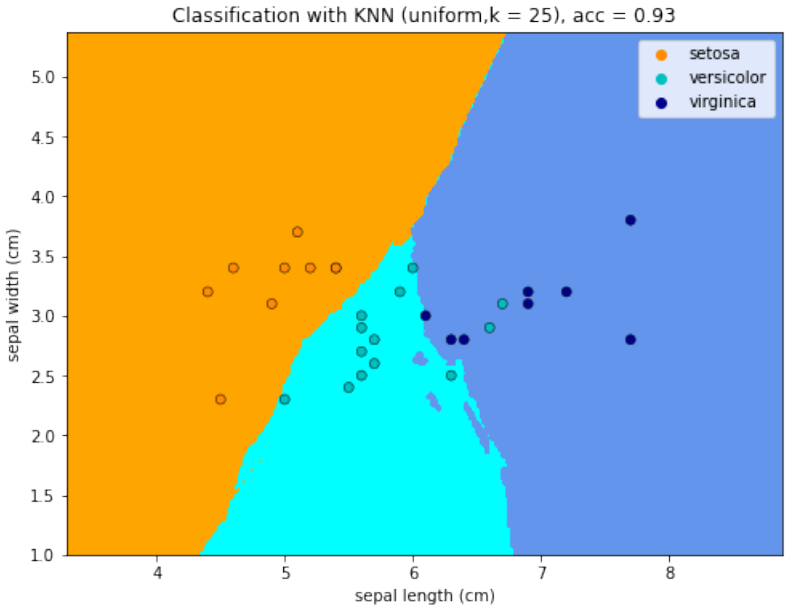

Figure 7: KNN best result with uniform weights and k = 25

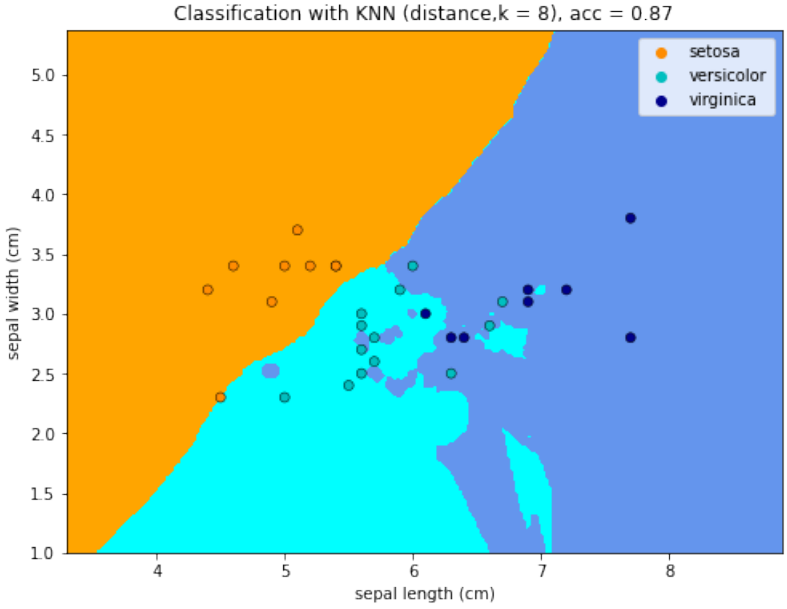

Figure 8: KNN best result with distance based weights and k = 8

c)

Compare the classification models for the iris data set that are generated by k-nearest neighbours (for the different settings from question 2b) and by logistic regression. Calculate confusion matrices for these models and discuss the performance of the various models.

In terms of performance, the models perform quite similarly. The KNN with uniform weights achieves the highest accuracy of 0.93 while logistic regression and KNN with distance weights achieves an accuracy of 0.87. Although, in terms of actual predictions the test set, the KNN with uniform weights "only" manages to correctly predict two observations compared to the other two models. In terms of the actual difference between logistic regression and KNN we can look at the decision boundaries from figures 5, 7 and 8. Logistic regression produces straight decision boundaries (since it has a linear classification rule), which makes it suitable when you have data that can be separated by straight lines or planes. Conversely, KNN uses the geometry of the feature space to classify points and thus can produce decision boundaries that capture more complex shapes and relationships of the data.

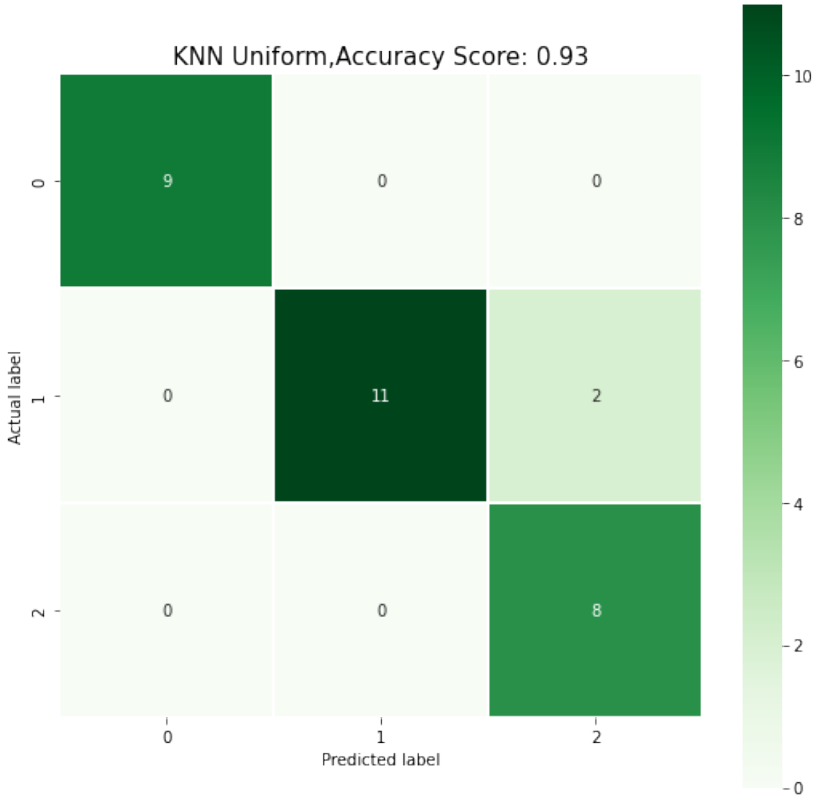

Figure 9: Confusion matrix for KNN with uniform based weights

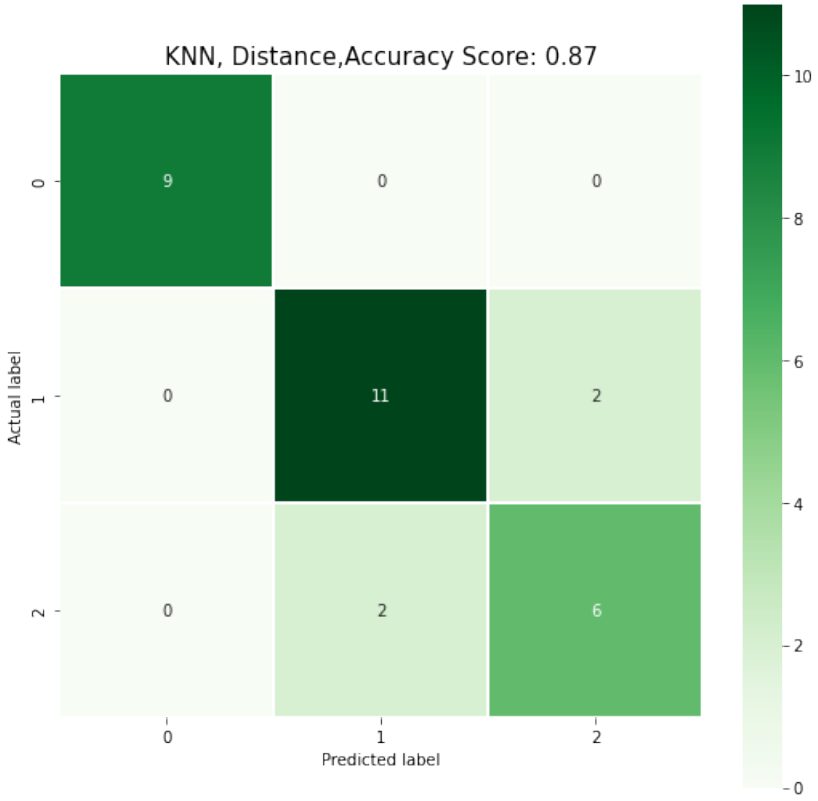

Figure 10: Confusion matrix for KNN with distance based weights

3

Explain why it is important to use a separate test (and sometimes validation) set.

The goal of constructing models in data science is to make predictions that can be generalized. In many cases we want a model that can make predictions beyond the scope of our data. For this we need to test our model on "unseen" data, also referred to as testing data. This is data that we haven't included in the training stage of our model, and thus when running our model on the testing set we get an indication of the predictive capabilities of our model. The problem is that we often have limited access to more data than we already have, and a simple solution is to split our existing data into a training and testing set. We train our model on the training set and then predict on the testing set. Sometimes we can also include a validation set which is used together with the training set to fine-tune the parameters of our model. This means that we train

and predict on the training and validation set to produce the most "accurate" model and once the model and its parameters are set, we do a final prediction on the testing set.