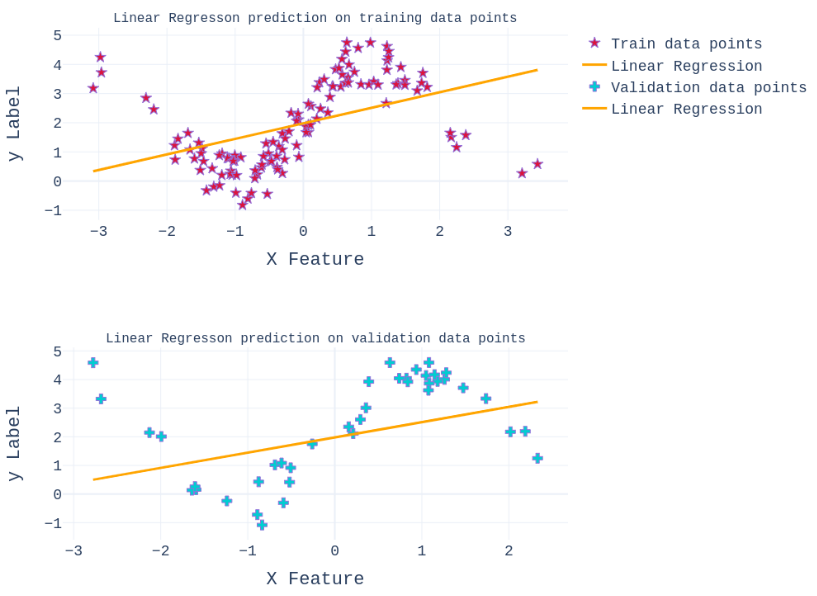
Exercise Sheet 4 – Machine Learning

INF161 Autumn Semester 2021

- Deadline: 1/10/2021 , 23:59
- Format: Your answers are to be returned in a single pdf report. You can also return scanned pages for your calculations. For results, your answers must include any values that are requested in the Notebook.

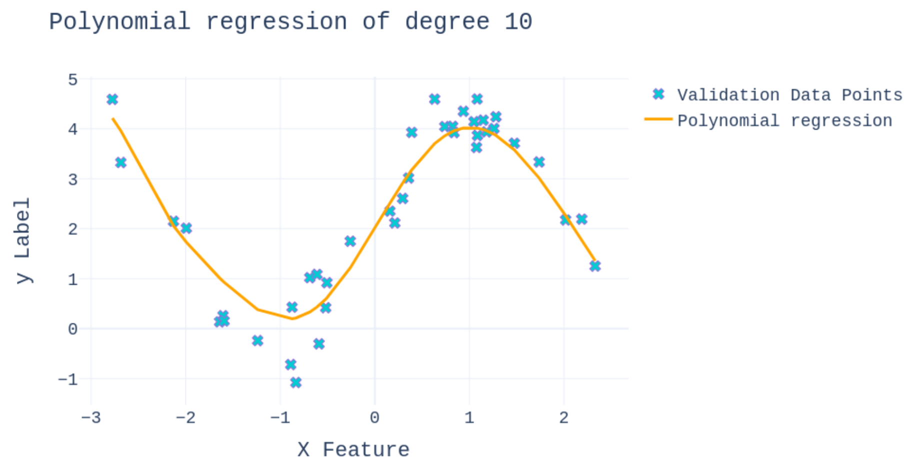
Exercise 1 *Regression*

Figure 1: Linear regression on the dataset



- *Load the* ***"regression_nonlin.csv"*** *file into a dataframe.*
- *Divide the dataset into three sets of training, validation, and testing with corresponding proportions of* ***60%***, ***20%***, *and* ***20%***, *respectively using the* ***train_test_split*** *function from the sklearn package. Set the initial random state to a specific initial seed (So that dataset is split into the same splits every time you run the notebook).*

Figure 2: Example figure of Polinomial Regression on the dataset



- *Fit a linear regression model to the training data.*
- *Plot the training and validation datasets along with the regression line and compute the Mean Square Error on both datasets. Your figure should look like figure 1.*
- *For each of the degrees [2, 5, 10, 20, 25], Fit a Polynomial Regression model of the chosen degree on the training dataset. Plot the results on the validation data along with the model prediction line like the example in figure 2 for each of the models.*
- *Compute the Mean Square Error on training and validation datasets for each degree and compare it to simple linear model. Does any of your models overfit or underfit?*
- *How well does your best performing model on validation data, make predictions on the test dataset?*

Exercise 2 *Classification*

- *Iris is a small dataset consisting of 150 vectors describing iris flowers, split into three different classes representing three species of the iris family. Each vector comes with a label (the name of the species) and a set of four features which are measurements of different parts of the flower. Those measurements tend to differ between the different species, thus it is possible to predict the species of an iris flower represented by a set of features. Load the Iris dataset directly from sklearn. You can alternatively download the dataset by clicking here.*
- *Store* ***the first two features (sepal length and sepal width)*** *in a matrix X. Also store the labels in a vector Y.*
- *Divide the dataset into three sets of training, validation, and testing with corresponding proportions of* ***60%***, ***20%***, *and* ***20%***, *respectively using the*

train_test_split *function from the sklearn package. Set the initial random state to a specific initial seed (So that dataset is split into the same splits every time you run the notebook).*

- *Perform a k-NN classification of your dataset for each k in 1, 5, 10, 20, 30. You can for instance use the* ***KNeighborsClassifier*** *class from sklearn.*
- *Use the Contour Plots from Plotly to visualize the predictions of the* ***KNeighborsClassifier*** *on a 2-dimensional grid for the values in the range of the* ***sepal length and sepal width*** *with the training points overlayed for every k (since there are three classes, you will need three different colors); To create the 2-D grid you can use the function* ***numpy.meshgrid*** *(documnetation and example here). The Hexcode of colors of the background for contour plot are: ['#f2ceb2', '#b4e1d4', '#b2d4e7'] and point colors are ['#0173b2', '#de8f05', '#029e73'].(You are free to choose your own colors as long as it is visible)*

  Your plots should look like the example in Figure. 3

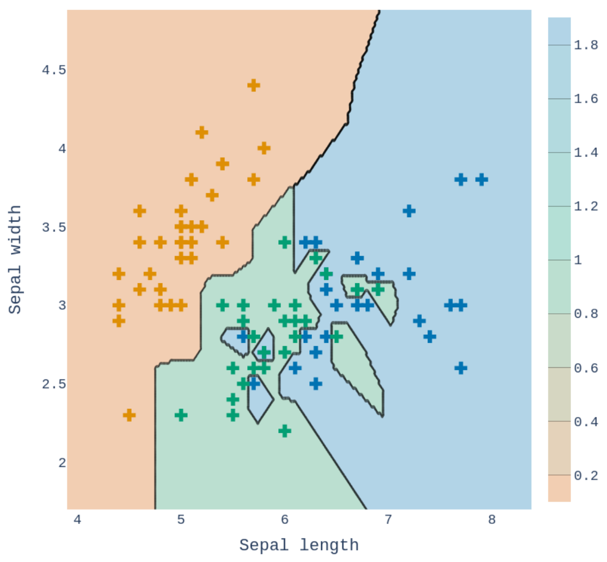


Figure 3: Example contour plot figure of K-NN Classifier outputs on a two dimensional grid of features from the Iris dataset

- *Plot the lines representing the change in training and validation accuracy as a function of different K (number of neighbors).*
- *How well does your best performing model on validation data, make predictions on the test dataset?*

Exercise 3 ***Information Leakage***

- *The script* ***CrossVal.py*** *first generates a dataset for classification based on random normal distribution. Then a routine cross validation process is* *applied on the dataset to select the best setting for number of most effective features which are fed into the **LogisticRegression** model to predict the class. Run the script and generate the orange1 line in Figure. 4.*

- *Given the fact that the data is randomly generated, it is expected that the model will perform randomly (around 50% accuracy like the blue line). Explain what is the incorrect part in the script that makes the model perform so well on randomly generated data? (hint: you can look up "Leakage" concept in machine learning)*

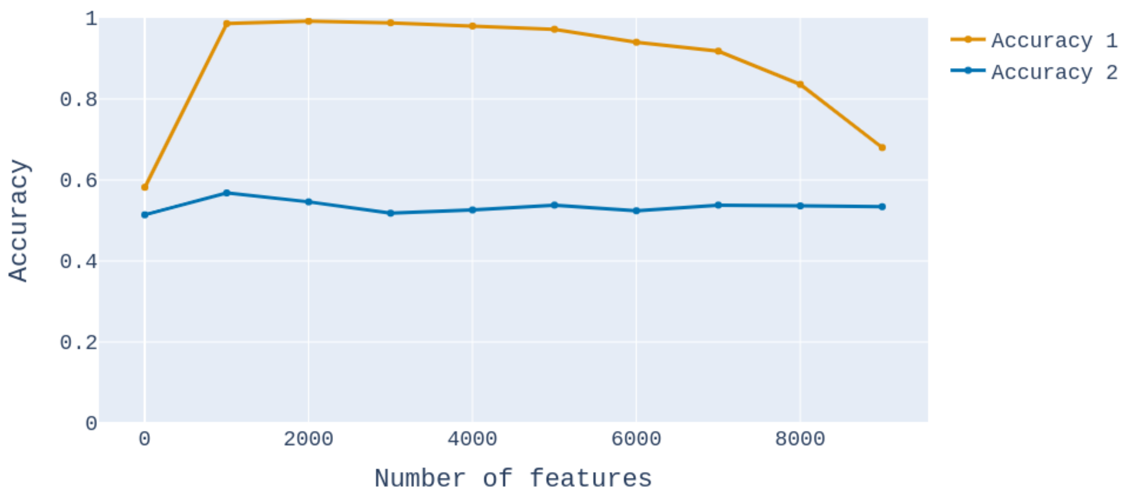


Figure 4: Accuracy of different options for number of selected features

- *Utilize **make_pipeline** function from **sklearn.pipeline** to create the correct cross-validation procedure for the classifier and plot the blue accuracy curve and compare it with the "wrong" version.*