Problem 1 (Statistical Learning) (10 points)

1. We are asked to inspect the bias-variance trade-off for some unknown model based on the Figure 1 below.

   (a) Describe what happens when we slowly increase the flexibility from 1 to 7. (1 point)

   (b) Considering only natural numbers, which flexibility would you recommend? Why? What is the MSE of the model you chose? (2 points)

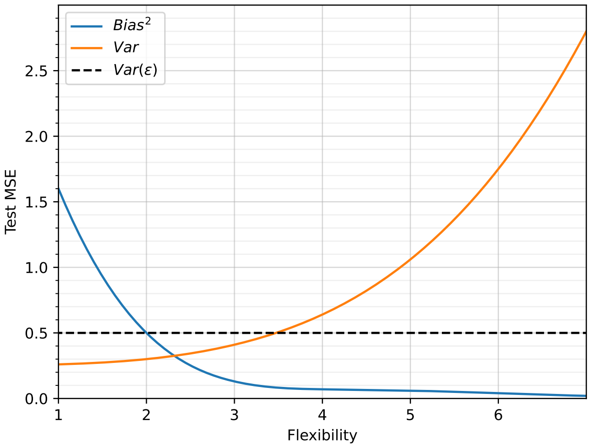


Figure 1: Test Mean Square Error (y-axis) for models of different flexibility (x-axis).

2. Explain for each of the three settings below what will happen, in terms of bias and variance, when we make the proposed change to the learning procedure. (3 Points)

   - We use local regression and change the fraction of training points from $s = 0.01$ to $s = 0.2$.
   - We replace the LDA classifier with the QDA classifier.
   - We use k-means clustering, and change the number of clusters k from 3 to 7.

3. In linear regression, we *generally* assume that the error (noise) term is zero on average. Why? (1 Point)

4. Are the following methods parametric or non-parametric? For each, explain why. (3 Points)

   - KNN (K-Nearest Neighbours).
   - SVM with a Radial Kernel.
   - A fully connected feed forward neural network of 3 layers of 25 nodes each, using standard sigmoidal activation functions.

Solution.

1. (a) As we increase the flexibility of our model, its bias decreases, while its variance increases. Initially, bias decreases faster than variance increases, which results in an overall decrease of the MSE. In the limit bias tends to 0, while variance keeps on increasing, which beyond a Flexibility of 3 results in models with ever-increasing MSE.

   (b) Flexibility of 3, which has $MSE = Var + Bias^2 + Var(\epsilon) = 0.4 + 0.14 + 0.5 = 1.04$. In contrast, flexibility 2 and 4 have much higher MSEs (resp. $0.3 + 0.5 + 0.5 = 1.3$ and $0.63 + 0.08 + 0.5 = 1.21$).

2. - When increasing the fraction of points considered, we increase the amount of data considered per prediction. Hence, reducing variance and increasing bias.
   - LDA assumes a common covariance matrix for all K classes whereas QDA assumes that each class has its own covariance matrix. Hence, bias goes down and variance up.
   - By increasing the number of clusters, there will be more ways to cluster the data, hence increasing variance and decreasing bias.

3. If we would *not* assume noise to be centered around zero, there would be infinitely many functions and noise distributions that all fit the data equally well. Only one of these would be unbiased, but no matter how much data we would have, we would not be able to tell which one this is. If we *do* assume zero-centered noise, we can *in the limit* identify the one unbiased function and noise distribution that generates the data.

4. - Non-parametric. There is no fixed number of parameters to describe the fitted model. The model is directly dependent on the used data.
   - Non-parametric. In the kernel we compute the pairwise distance between all training points, which defines our decision boundary.
   - Parametric. Our NN can be described with a fixed finite number of parameters.

PROBLEM 2 (REGRESSION) **(10 points)**

We are asked to fit a quadratic polynomial (degree = 2) to the data from the following table. We are told that $\beta_0 = 0$ and hence do not have to estimate it.

X_1	-1	-0.7	-0.3	0	0.3	0.7	1
Y	4.09	1.33	0.99	-0.46	0.44	1.75	3.41

Recall that a quadratic polynomial (with fixed $\beta_0 = 0$) takes the form $Y = \beta_1 X_1 + \beta_2 X_1^2$, but that it is often convenient to formulate it as $\mathbf{X}\beta = \mathbf{Y}$ with $\beta = [\beta_1\ \beta_2]^\top$ and $\mathbf{X} = [\mathbf{X_1}\ \mathbf{X_1^2}]$. Note that the square in the last formula denotes the element-wise squaring of $\mathbf{X_1}$.

1. Estimate the coefficients β_1 and β_2 using the following result
$$\left(\mathbf{X^T}\ \mathbf{X}\right)^{-1} = \begin{bmatrix} 0.32 & 0 \\ 0 & 0.4 \end{bmatrix}.$$
Explain each step. (2 points)

2. Using an advanced fitting procedure, Prof. Vreeken obtained $\hat{\beta}_1 = -0.1$ and $\hat{\beta}_2 = 3.5$. Based on these estimates, what can you say of the relationship between X_1 and Y? Do you think a linear model (without the quadratic term) would give a better or worse fit, or is that impossible to say? Why? (2 point)

3. Sketch the residual plot for the given data, using the estimates provided in Problem 2.2. Interpret your plot. Does the plot support your conclusion from Problem 2.2? (2 points)

4. Compute the R^2 score, using again the parameters provided in Problem 2.2. What does the R^2 value, in general, tell you about the fit of your model? (2 points)

5. If we know that the data was generated as $Y = \beta_1 X_1 + \beta_2 X_1^2 + \epsilon$, with $\mathbb{E}[\epsilon] = 0$ and $Var(\epsilon) = 0.15$, can you tell, 95%-confidently, that the trend given by β_2 in Problem 2.2 holds? Why? (2 points)

Solution.

1. We have to compute $[\beta_1\ \beta_2]^T$ from $((X^TX)^{-1}X^T)Y$ (see Linear Regression I slides, slide no. 24). We are given $(X^TX)^{-1}$. Thus, we first multiply by X^T:

$$\begin{aligned}&(X^TX)^{-1}X^T\\ &= \begin{bmatrix}0.32 & 0\\ 0 & 0.4\end{bmatrix}\begin{bmatrix}-1 & -0.7 & -0.3 & 0 & 0.3 & 0.7 & 1\\ 1 & 0.7^2 & 0.3^2 & 0 & 0.3^2 & 0.7^2 & 1\end{bmatrix}\\ &= \begin{bmatrix}-0.32 & -0.224 & -0.096 & 0 & 0.096 & 0.224 & 0.32\\ 2 & 0.4 & 0.196 & 0.036 & 0 & 0.036 & 0.196\end{bmatrix}\end{aligned}$$

Now we multiply this matrix by Y to get:

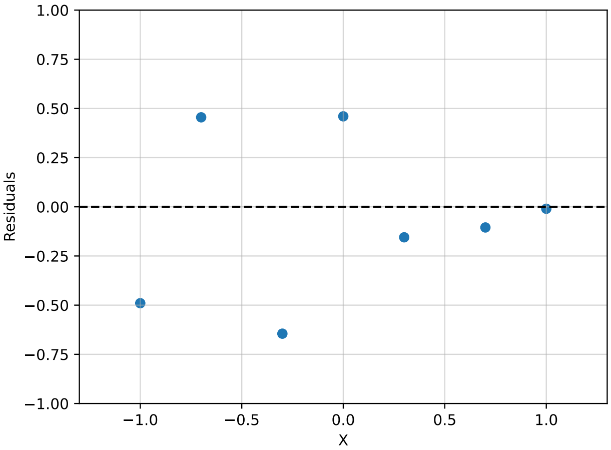
$$\begin{aligned}&\left((X^TX)^{-1}X^T\right)Y\\ &= \begin{bmatrix}-0.32 & -0.224 & -0.096 & 0 & 0.096 & 0.224 & 0.32\\ 2 & 0.4 & 0.196 & 0.036 & 0 & 0.036 & 0.196\end{bmatrix}\begin{bmatrix}4.09\\ 1.33\\ 0.99\\ -0.46\\ 0.44\\ 1.75\\ 3.41\end{bmatrix}\\ &= \begin{bmatrix}-0.17632\\ 3.65516\end{bmatrix} = \begin{bmatrix}\hat{\beta}_1\\ \hat{\beta}_2\end{bmatrix}\end{aligned}$$

Thus the result, rounded to two decimals, is $\hat{\beta}_1 = -0.18, \hat{\beta}_2 = 3.66$.

2. If the data would be linear, we would obtain an (almost) zero value for β_2. The fact that we here have a low value for β_1 and high value for β_2 indicates that the relationship is not linear, but (at least) quadratic. Or, in other words, a linear model would not be able to fit this data as well as a quadratic model.

3. Yes, the residual plot shows no discernible pattern (e.g. a U-shape), and thus supports the conclusion, that the data follows a quadratic polynomial:

X_1	-1	-0.7	-0.3	0	0.3	0.7	1
Residual	-0.49	0.455	-0.645	0.46	-0.155	-0.105	-0.01



4. The R^2 score tells us the proportion of the variance in the data that is explained by our model. From slide set Linear Regression I, slide 21, we get the formula $R^2 = 1 - \frac{RSS}{TSS}$. The RSS is computed as $\sum_{i=0}^{n}(y_i - \hat{y})^2 = 1.028$, and the TSS as $\sum_{i=0}^{n}(y_i - \bar{y})^2 = 15.515$. Thus we get $R^2 = 1 - \frac{1.028}{15.515} = 0.933$.

5. We compute the interval $[\beta_2 - 2 * SE(\beta_2), \beta_2 + 2 * SE(\beta_2)]$. According to the slides, and using $\sigma^2 = 0.15$, we get $SE(\beta_2)^2$ as $\frac{\sigma^2}{\sum_{i=0}^{n}(x_i^2 - avg(x^2))^2} = 0.14$. Thus, $[\beta_2 - 2 * SE(\beta_2), \beta_2 + 2 * SE(\beta_2)] = [2.75, 4.25]$. Since all values in this interval are clearly above zero, we conclude that the quadratic trend holds.

Problem 3 (Non-linear) **(10 points)**

1. Draw a decision tree that produces the partitioning shown in Figure 2. (2 points)

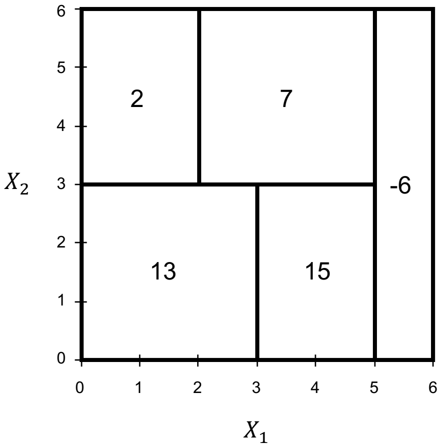


Figure 2: Partition produced by an unknown decision tree. The values within a region indicate the mean of Y within that region.

2. We consider regression splines with one predictor. (2 points)

   How many degrees of freedom has a regression spline model with two knots ($K = 2$) and linear functions ($d = 1$) as splines, when we as usual enforce continuity in derivatives at each knot up to degree $d - 1$. Explain the interpretation for each degree of freedom. **Do not just use the equation you might happen to know.**

   Example explanation of how each degree of freedom of a linear regression model can be interpreted: "The first degree of freedom specifies the intercept, the second specifies the slope of the linear function."

3. Suppose that we compute a curve $\hat{g}$ to smoothly fit a set of n points using the following formula:

$$\hat{g} = \arg\min_{g} \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g'''(x) \right]^2 dx \right) \quad .$$

To flex his coding skills Prof. Vreeken implemented a fitting algorithm that allows values of λ of i) $\lambda = 0$, ii) $\lambda = 1$, and iii) $\lambda = \infty$. He applied the algorithm on some data and got the result shown below. Which of the three possible values of λ was used to obtain the fit shown in Figure 3? Explain your answer. (2 points)

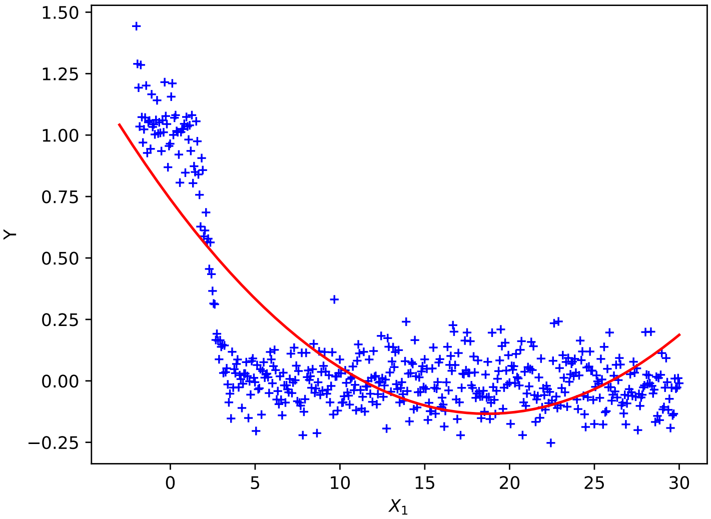


Figure 3: Smoothing spline with unknown parameter λ

4. To prepare for EML next year, Prof. Vreeken is studying an obscure regression method that is similar to Ridge Regression and LASSO, but has a different penalty term. It estimates $\hat{\beta}$ by minimizing the following term

$$RSS + \lambda \sum_{j=1}^{p} I(\beta_j \neq 0) \ ,$$

where $I(\beta_j \neq 0)$ is an indicator function that takes on a value of 1 if $\beta_j \neq 0$ and 0 otherwise. How does this method relate to best subset selection? (1 point)

5. Prof. Vreeken wants to fit a function to the data shown in Figure 4. He consider two models.

   As the first model (Model A) he considers a Generative Additive Model (GAM) with cubic polynomials (degree = 3) as basis functions

   $$y_i = \beta_0 + \sum_{j=1}^{2} f_j(x_{ij}) \ ,$$

   where f_j is a degree 3 polynomial. As the second model (Model B), he considers a plain linear model over X_1, X_2, and the combination $X_1 * X_2$.

   $$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i1} x_{i2}$$

   Explain why one of the two models can fit the data well, while the other one cannot.

(2 points)

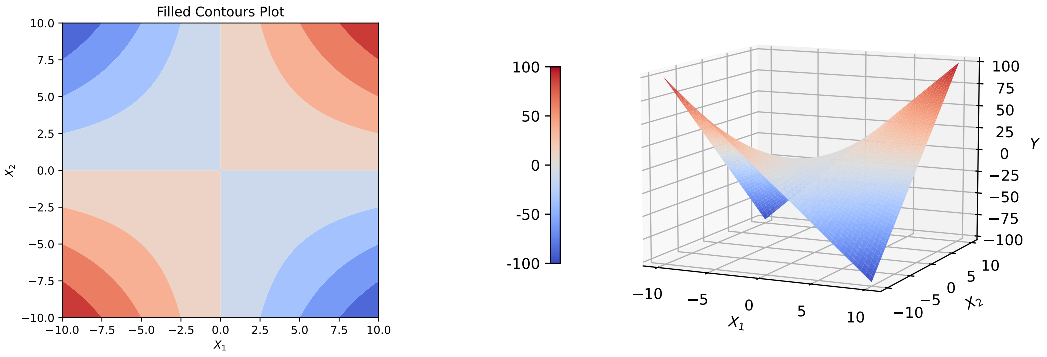


Figure 4: Same data visualized in two different ways. [Left] Contour plot where Y is indicated by the color, from deep blue representing Y values between -75 and -100 to dark red representing Y values from 75 to 100. [Right] 3D Plot of the same function. Y value shown on the vertical axis.

6. We want to learn a Linear Model Tree from the data shown in Figure 5. A Linear Model Tree is very similar to a standard regression tree. It is only different from a standard regression tree in that $\hat{y}_{R_k}$ now represents a linear model instead of the mean.

   At each step, the algorithm chooses that predictor X_j and cut point s, creating two new regions $R_1(j,s) = \{X|X_j < s\}$ and $R_2(j,s) = \{X|X_j \geq s\}$ solving

$$\min_{j,s} \sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2 , \tag{3.1}$$

   where $\hat{y}_{R_k}$ denotes the prediction of a linear model fitted using the data from that region. That is, $\hat{y}_{i,R_k} = x_i a_{R_k} + b_{R_k}$.

   Sketch[1] into Figure 5 how a Linear Model Tree would fit the shown data. Split the data into **at least** 3 and **at most** 5 regions. The resulting Linear Model Tree does not have to be balanced. Clearly indicate the boundaries of the region as well as $\hat{y}_{R_k}$ of each region. (1 point)

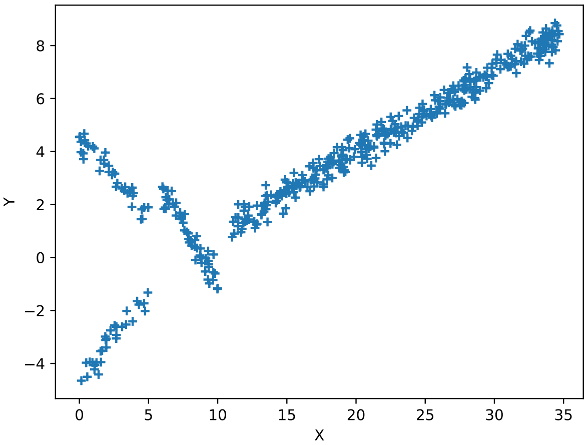
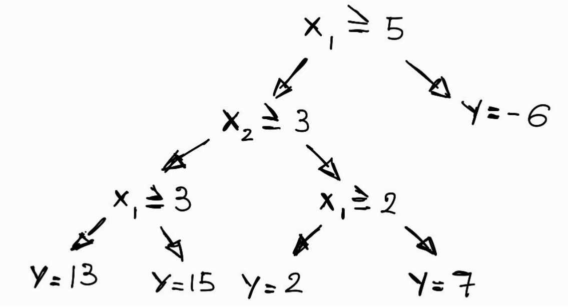


Figure 5: Data for the Linear Model Tree.

[1] You may also recreate the plot by hand, but do make sure you include all key details.

Solution.

1. The partition shown in Figure 2 corresponds to the following tree:



2. Since we use linear splines ($d = 1$), we have 2 parameters for each region (slope and intercept). Since there are $K = 2$ knots, we have $K + 1 = 3$ regions. Therefore, we have $2 * 3 = 6$ parameters in total.

   As we need to impose continuity in both knots, we include two constraints to the model, reducing by 2 the degrees of freedom. Hence, we have in total $6 - 2 = 4$ degrees of freedom.

   The 4 degrees of freedom can be interpreted as the slopes and intercepts of the linear models in each of the outer regions, since the parameters of the linear model in-between need to be selected to connect the two points at the knots to ensure continuity.

3. Since the fit of the data in Figure 3 has a quadratic shape, we need to set λ to ∞.

   If were to set λ to 0, there would be no regularization term and the model would hence try to fit the data perfectly, resulting in a very "jumpy" spline. If we were to set λ to ∞, we penalize so extremely strongly that we would disallow any model with a non-constant curvature change, $g'''(x) \neq 0 \; \forall x$. This means that for $\lambda = \infty$ only models with a constant curve change, such as a quadratic function (linear curvature) or a linear function (zero-curvature) would be allowed, as $g(x) = \int\int\int g'''(x) d^3x = \int\int\int 0 d^3x = \beta_0 + \beta_1 x + \beta_2 x^2$. If we were to set $\lambda = 1$ we would balance between fitting the data and having a quadratic form. Since the fit of the data in Figure 3 has a quadratic shape, we need to set λ to ∞.

4. **Long answer:** Each indicator $I(\beta_j \neq 0)$ in the penalization term acts as a binary variable that reads "We select feature X_j corresponding to coefficient β_j." Selecting a feature adds a penalty of λ, and by solving the optimization problem we find a trade-off between selecting more features, and making sure these are informative enough to reduce the RSS loss.

   Short answer: The formulation given is the unconstrained version (using Lagrange multipliers and removing constant terms) of the best subset selection problem, as introduced in Lecture 7 (slide 23).

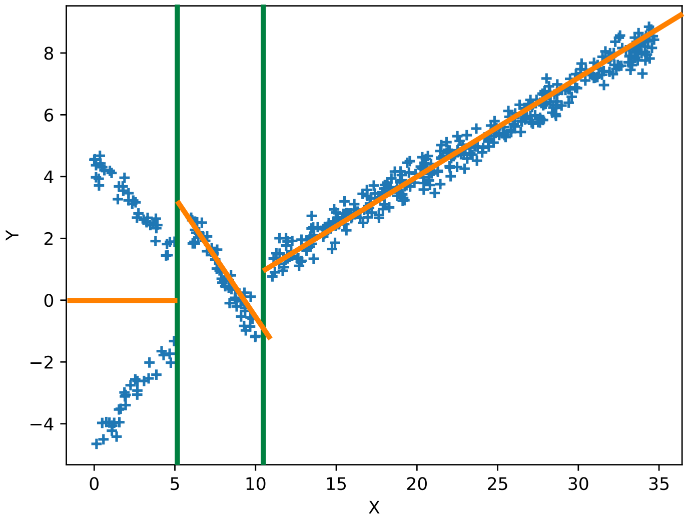
5. While the GAM has more degrees of freedom, it cannot properly fit the data shown in Figure 4. This is because the model is still additive with respect to the features, and thus cannot model higher-order interactions, which we can observe in the contour plot (values depend on how far x_1, x_2 are from zero, and the quadrant they lay on).

   The linear model that includes $X_1 * X_2$ as a feature is able to model a multiplicative interaction between X_1 and X_2. From the contour plot we see that the function is $Y = X_1 * X_2$. Therefore, a linear model that includes this interaction term will be able to fit the data perfectly.

6. The solution is shown below. The two green lines divide the space into three clear regions. The orange lines describe the linear regressor of each region.

   The first and second split will be at $X \approx 5$ and $X \approx 11$. Which out of these is the actual first, and which is the actual second split is unclear to the eye and would require computing the actual RSS. We do not require a third split because this would not improve the RSS much.

   Note that we cannot split on Y.



PROBLEM 4 (CLASSIFICATION) **(10 points)**

1. In classification, values outside of the typical distribution of a predictor but on the "correct" side of the decision boundary are a special kind of outliers. Are the following methods sensitive to these special outliers? Explain why (not). (2 points)

   - Logistic regression.
   - Classification Decision Trees.
   - Support Vector Machines (SVM).
   - K-Nearest Neighbours (KNN).

2. You are given the following decision boundary of a Support Vector Classifier

   $$7 - 3x_1 - 2x_2 + 4x_3 + 7x_4 \ .$$

   Using this boundary, assign the following point to either the positive or negative class, $x = (-2, 3, 6, -4)$. (1 point)

3. Recall that according to the Bayes Theorem, $Pr(Y = k|X = x) \propto \pi_k * f_k(x)$, where π_k is the prior and f_k the likelihood function. A domain expert tells you that their data most certainly follows a Poisson distribution with distinct values λ_k for each of the K classes, where

   $$f(x, \lambda_k) = \frac{(\lambda_k)^x \, e^{-\lambda_k}}{x!} \ .$$

   Derive the discriminant function. Simplify the discriminant function as much as possible. (2 points)

4. We train a decision tree for a binary classification problem. We are given 20 data points out of which 16 belong to Class 1 and 4 to Class 2. Prof. Vreeken's implementation finds only one possible way to split the data: one region containing 10 points out of which 10 belong to Class 1, and a second region containing 10 points out of which 6 belong to Class 1. Does it make sense to split the data this way? Why (not)? How does this setting relate to Misclassification Error, Gini Index, and Cross Entropy? (2 points)

5. We train a decision tree for binary classification using the Gini Index as the quality measure. We are given a categorical predictor with q unordered values.

   (a) Show that there are $2^{q-1} - 1$ possible partitions into two groups, where each group has at least one element. (2 point)

   (b) We want to avoid testing all $2^{q-1} - 1$ possible partitions. How can we nevertheless find the best partition? In other words, how can we find an order over the values that allows us to find the best partitioning. (1 point)

Solution.

1. - Logistic regression: No, the decision boundary is mostly defined by points closest to the decision boundary.
   - Decision trees: No, the splitting is determined based on the sample proportions in each split region and not by absolute values.
   - Support vector machine: No, it is only dependent on support vectors and does not 'care' about samples on the correct side of the margin at all.
   - K-nearest neighbours: No, it performs the majority voting of k nearest neighbours. An outlier has the same vote weight as other points and it is on "correct" side of decision boundary.

2. To classify a datapoint we plug the given values, x_1, x_2, x_3, x_4, into the decision boundary and compute its result.

$$7 - 3 * (-2) - 2 * 3 + 4 * 6 + 7 * (-4) = 3$$

   Since $3 > 0$ the sample belongs to the positive class.

3. Given that $P(Y = k \mid X = x) \propto \pi_k * f_k(x)$, we can simply plug-in our calculated π_k and the given $f_k(x)$ in to the equation of the Bayes Theorem. Then, by taking $\arg\max_k$ of the product $\pi_k * f_k(x)$ we approximate the Bayes optimal decision.

$$\begin{aligned}
\arg\max_k \quad p_k(x) &= \arg\max_k \quad \pi_k \frac{(\lambda_k)^x e^{-\lambda_k}}{x!} \\
&= \arg\max_k \quad \log(\pi_k) + \log\left((\lambda_k)^x\right) + \log\left(e^{-\lambda_k}\right) - \log(x!) \\
&= \arg\max_k \quad \log(\pi_k) + \log\left((\lambda_k)^x\right) - \lambda_k - \log(x!) \\
&= \arg\max_k \quad \log(\pi_k) + x\log(\lambda_k) - \lambda_k - \log(x!) \\
&= \arg\max_k \quad \log(\pi_k) + x\log(\lambda_k) - \lambda_k
\end{aligned}$$

4. Yes it still makes sense to split the data. As in the one case we can be much more certain of our predictions, in other words it increases the node purity. The considered split does not improve the classification error, however it does improve the Gini Index and Entropy scores.

5. (a) We can represent a split as a binary number of length q where a 1 at index i means we place datapoints where our predictor has value q_i in the right hand split and a 0 means we place that datapoint in the left hand split. This gives us 2^q possible splits. Note that each split can be represented in two different ways, for example `01001` and `10110` represent the same split. To not double count splits, we divide by 2 giving us 2^{q-1} splits. Finally, as we do not allow empty splits we have to subtract one from this number, which results in a total number of $2^{q-1} - 1$ possible splits.

(b) By ordering the q values according to the proportion of data points falling into class 1 and then splitting as if it was an ordered predictor (See Lecture 12, Slide 33.).

Problem 5 (Clustering and Dimensionality Reduction) **(10 points)**

Local French bakery "Les Deux Croissants" asked us to analyse their data.

1. We first consider the data given in Figure 6.

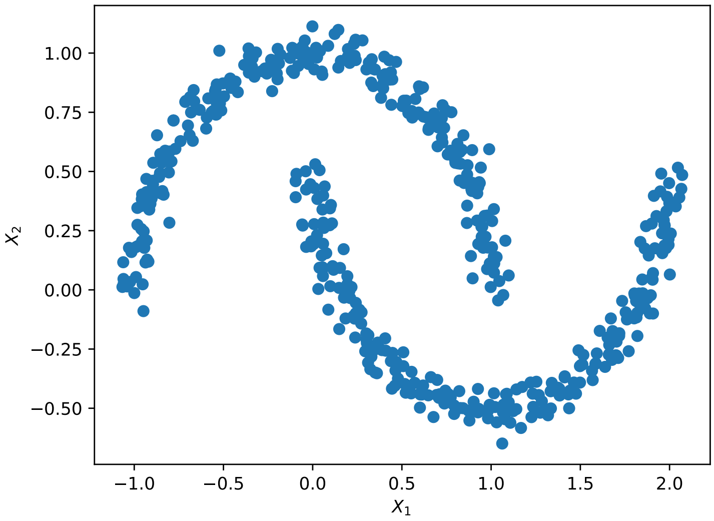


Figure 6: Data from bakery "Les Deux Croissants".

(a) Sketch into Figure 6 the clustering that k-means with $k = 2$ and using Euclidean distance is most likely to find. Explain why k-means is a good/bad choice for clustering this data. (2 points)

(b) Which linkage measure would you recommend for hierarchically clustering this data? Why? (1 point)

(c) Clustering high dimensional data is hard. It may help to first reduce the dimensionality, and then cluster. Consider the following two approaches for the data shown in Figure 6.

- Use PCA to reduce the dimensionality to one dimension, and then apply k-means with $k = 2$.
- Use t-SNE to reduce the dimensionality to one dimension, and then apply k-means with $k = 2$.

Describe the expected result for each. Which one would you choose and why? (2 points)

French bakers are very concerned with the butteriness of their croissants. They wish to predict this value, but although they are certain that only few predictors truly matter, they cannot agree which these possibly would be.

2. Suppose we are given $n = 1000$ datapoints and $p = 20$ predictors. Y is a linear function of 3 predictors. We consider using either PCR or PLS to reduce the dimensionality of this data to 5 dimensions. Describe a scenario in which PLS would work but PCR would fail to provide a meaningful result. (2 points)

At the last moment, the baker's assistant raises a serious concern about Problem 5.1a above. Does k-means even converge?

3. Explain why the k-means algorithm always converges. (2 points)

4. Is k-means, in general, sensitive to outliers in the data? Explain why (not). (1 point)

Solution.

1. (a) k-Means with Euclidean distance assumes spherical clusters. The clusters shown in Figure 6 are clearly not spherical, the “spheres” of the individual clusters overlap and hence k-means would fail to group the data per “croissant”.

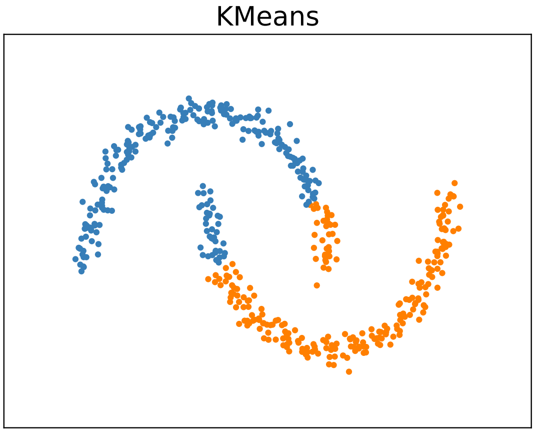


Figure 7: k-Means clustering of data from bakery “Les Deux Croissants”. Reported clusters by k-means are represented by color.

(b) Single linkage, as the closest neighbour of each datapoint belongs to the same cluster. Hence we slowly “grow” our clusters by adding that point (or clusters of points) that is closest to any point already in our cluster.

(c)
- PCA takes a global view on the data, maximizing variance along the first principle component. Thereby it is not able to preserve the cluster structure, i.e. there will be a large overlap between these clusters in the 1 dimensional projection. k-Means would hence result in very impure clusters.
- t-SNE preserves local distances, i.e. essentially giving up on the shape of the cluster in order to preserve local distances. For this dataset, this would result in two distinct clusters, corresponding to the two croissants, by which k-means would be able to correctly cluster the data. i.e. all points of one croissant would be in the same cluster.

Clearly, for this data t-SNE is the better choice as it allows us to cluster the data “correctly”.

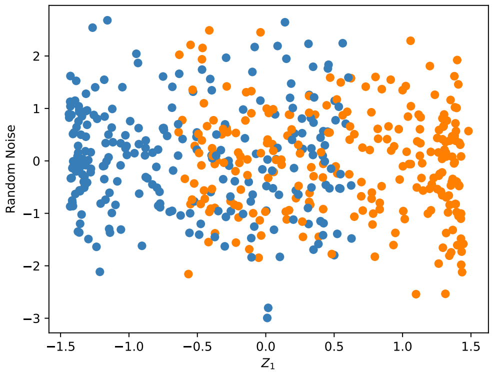


Figure 8: Jitter plot[1] of datapoints projected onto the first principal component. Blue points belong to upper "croissant" and orange points to lower "croissant".

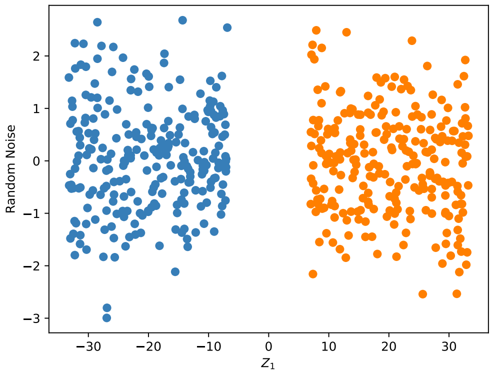


Figure 9: Jitter plot of t-SNE embedding into one dimension. Blue points belong to upper "croissant" and orange points to lower "croissant".

2. **Option 1:** In a scenario where Y is generated by 3 predictors that together only account for a small fraction of the variance. This scenario only works if we do not standardize our inputs beforehand.

   Option 2: In a scenario where most of the features are not correlated to each other.

   Explanation: As PCR does not consider Y when reducing the dimensionality, it has no incentive to maintain any of the information about Y that is contained in these 3 predictors. PLS, on the other hand, does place most weight on those predictors that are most correlated with Y, and thereby it does preserve the information about Y that is contained in these 3 predictors. This will overall result in better performance when using PLS.

[1] Jitter plot: As we reduce the dimensionality to one, all datapoints are be distributed along one line. For a better visual interpretation we sample for each datapoint a random value and plot these pairs. Importantly the random "y" value is only used for plotting.

3. By the “k-means algorithm” we implicitly refer to Lloyd’s algorithm, which starts with a set of k initial centroids and alternates between two steps; in the first step of each iteration it assigns points to the nearest centroid, whereas in the second it recomputes the centroids as the mean of all points assigned to it. Convergence is achieved when the assignments stop changing.

   Intuitively, k-means converges because at each step the objective function (strictly) decreases and is lower bounded by 0, since it is positive. Furthermore, it converges in a finite number of steps, since the number of assignments is finite. Since our objective function decreases in each step ending up in a cycle is not possible.

4. The k-means algorithm is rather sensitive to outliers, due to the use of the mean as a centroid computation statistic, which is inherently sensitive to outliers. The effect may range from over-/under-estimating the true cluster center to selecting the outlier as a single cluster.