Machine Learning

Sebastian Gherhes - sag1g15

28018788

1 Data I: separate 2 Gaussians

LDA is a method developed by Fisher. It's purpose is to reduce dimensionality of the data and to classify the data. This classifying is done by choosing a vector onto which to project the data in order to maximize the ratio of between-class variance and within-class variance. (Scikit-learn n.d.)

For the first part of the coursework, one had to look at samples of data generated from two 2-dimensional Gaussian distributions : $x_a \sim N(x|m_a, S_a), x_b \sim N(x|m_b, S_b)$. The data points belonged to classes a and b. The mean vectors used were: $m_a : \begin{bmatrix}1.15 & 3.0\end{bmatrix}$ and $m_b : \begin{bmatrix}4.16 & 2.0\end{bmatrix}$ and the covariance matrices were: $\Sigma_a : \begin{bmatrix}1.0 & 0.5\\ 0.5 & 1.0\end{bmatrix}$ and $\Sigma_b : \begin{bmatrix}0.6 & 0.3\\ 0.3 & 0.7\end{bmatrix}$. There are 500 data points sampled from the first distribution and 500 data points sampled from the second distribution. Figure 1 shows the sampled data set.

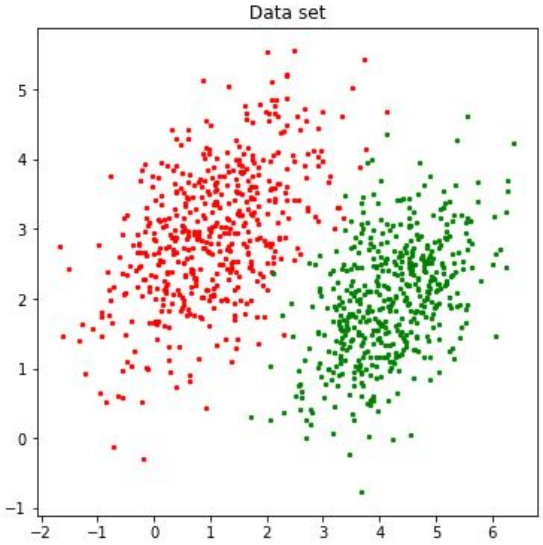
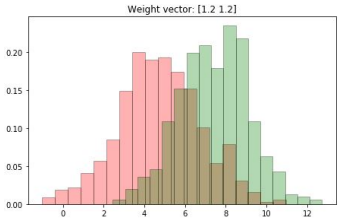


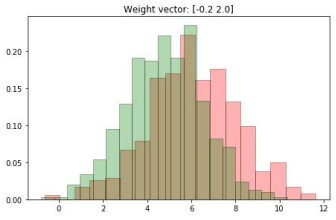
Figure 1: Sampled data set

For the first part, 3 random vectors were chosen to show an illustrative choice for the direction of w. The sampled data has been projected on all of those vectors. Figure 2 shows the histogram of the projection onto those vectors. Figure 2d shows the projection onto the optimal direction. The least amount of overlapping is also present in that data, meaning that the variance between the data is easily visible. In order to find this optimal direction w*, we had to maximise the Fisher ratio: $F(w) \triangleq \frac{(\mu_a - \mu_b)^2}{\frac{n_a}{n_a+n_b}\sigma_a^2 + \frac{n_b}{n_a+n_b}\sigma_b^2}$. To calculate the means μ_c and standard deviations σ_c^2 of the projected data, we have set $y_c^n \triangleq w * x_c^n$ for class label c ∈ a,b. After, we have chosen a starting weight vector $\text{w}(0) = \begin{bmatrix}1 & 1\end{bmatrix}$ which we have rotated by angles θ with the following rotation matrix: $R(\theta) = \begin{bmatrix}\cos\theta & -\sin\theta\\ \sin\theta & \cos\theta\end{bmatrix}$. Rotating the starting vector with this rotation matrix

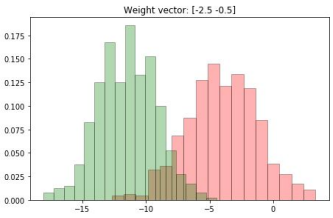
has given us w(θ) = R(θ)w(0), where θ has the value of each angle. As mentioned before, in order to find the optimal vector, we had to compute the maximum value of F(w(θ)) and find it's direction w*: w* = argmaxF(θ) = argmaxF(w(θ)).



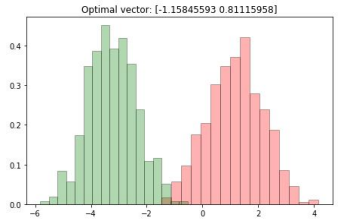
(a) Projection on direction: [1.2 1.2]



(b) Projection on direction [-0.2 2.0]



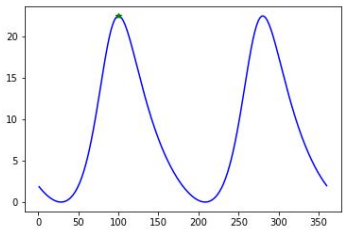
(c) Projection on direction: [-2.5 -0.5]



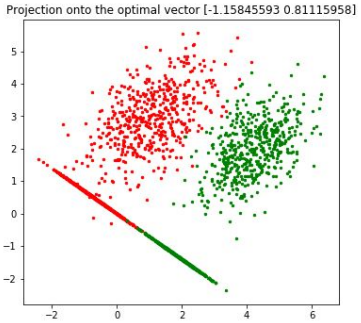
(d) Optimal direction: [-1.15845 0.81115]

Figure 2: Histogram of projection onto vectors

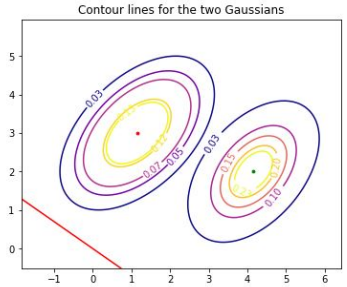
Figure 3a illustrates the plot of the Fisher ratio at each θ angle. The green star on the graph shows the maximum value for F(w) which gives use the optimal vector. Figure 3b shows the projection of the data on the direction w*, while Figure 3c plots the equi-probable contour lines for each class with the direction of the optimal vector.



(a) F(w) function at each rotating angle



(b) Projected data onto optimal vector



(c) Contour line for each class with the direction of the optimal vector

Figure 3: F(w) function, projected data onto optimal vector and contour lines

The decision boundary devised by Bayes' is given by the equation $P(c = a|X = x) = P(c = b|X = x)$. This means, the probability of class a given x is equal to the probability of class b given x. If for some point x, P(c = a|X = x) > P(c = b|X = x) $\rightarrow$ x is assigned to class a, otherwise x is assigned to class b. The decision boundary at equal posterior probability for either class is defined by the log-odds which need to be equal to 0: $log(\frac{P(c=a|X=x)}{P(c=b|X=x)}) = 0$. Looking at Bayes theorem, for class a: $P(c = a|X = x) = \frac{P(c=a|X=x)P(c=a)}{P(x)}$. Same theory applied for class b: $P(c = b|X = x) = \frac{P(c=b|X=x)P(c=b)}{P(x)}$. Looking at log-odds: $log(\frac{P(c=a|X=x)}{P(c=b|X=x)}) = log\frac{P(X=x|c=a)P(c=a}{P(X=x|c=b)P(c=b)}$. When we look at LDA, we take in the consideration the special case in which we

assume that the classes have a common covariance matrix and for that we have any class k = a,b :

$$f_k(x|c=k) = \frac{1}{2\pi\sqrt{det(\Sigma)}} exp(-\frac{1}{2}(x-\mu_k)^T \cdot \Sigma^{-1} \cdot (x-\mu_k)). \tag{1}$$

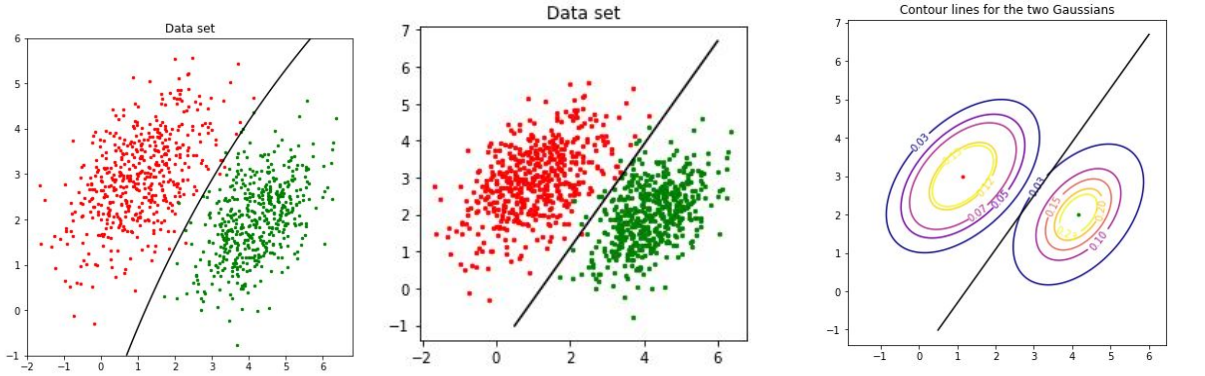
Therefore : $\frac{P(X=x|c=a)P(c=a}{P(X=x|c=b)P(c=b)} = log\frac{f_a(x|c=a)P(c=a)}{f_a(x|c=b)P(c=b)} = log(f_a(x|c=a) + log(P(c=a)) - log(f_b(x|c=b) - log(P(c=b)) = log\frac{f_a(x|c=a)}{f_b(x|c=b)} + log\frac{P(c=a)}{P(c=b)} = log\frac{P(c=a)}{P(c=b)} - \frac{1}{2} \cdot (\mu_a + \mu_b)^T \cdot \Sigma^{-1} \cdot (\mu_a - \mu_b) + x^T \cdot \Sigma^{-1} \cdot (\mu_a - \mu_b)$. The required log of ratio of probabilities is zero if:

$$\delta(x) = log\frac{P(c=a)}{P(c=b)} - \frac{1}{2} \cdot (\mu_a + \mu_b)^T \cdot \Sigma^{-1} \cdot (\mu_a - \mu_b) + x^T \cdot \Sigma^{-1} \cdot (\mu_a - \mu_b) = 0 \rightarrow \delta(x) = 0 \tag{2}$$

$\delta(x)$ is a linear function in x which is the equation that gives us the decision boundary for classifying our data. When the covariance matrices are not equal, the linear discriminant analysis becomes a quadratic discriminant analysis. Working out the maths as before gives us the following quadratic function:

$$\delta_a(x) = -\frac{1}{2}log|\Sigma_a| - \frac{1}{2}(x-\mu_a)^T\Sigma_a^{-1}(x-\mu_a) + log(P(c=a)) \tag{3}$$

The decision boundary between classes a and b is a quadratic function $x : \delta_a(x) = \delta_b(x)$. Figure 8a shows the QDA and the plotted deicision boundary for the sampled data, while Figure 8b shows the LDA and the plotted decision boundary for the sampled data. (Hastie et al. 2001)



(a) QDA of the sampled data (b) Sampled data set classified by the decision boundary (c) Contour line plot with the decision boundary

Figure 4: Decision boundary drawn on the sample data graph and the contour line graph

The consequences of using this formula for the discriminant $F_{unbalanced}(w) \triangleq \frac{(\mu_a-\mu_b)^2}{\sigma_a^2+\sigma_b^2}$ is that it does not take into account the different fractions of data in each class. The data becomes homogeneous, it is not as accurate as the previous one, it becomes unbalanced. If, for example, the sample sizes were of a different size for class a and b, the variance of the class with the higher sample would be bigger than the variance of the class with the smaller sample. This is taken into account in the Fisher ratio F(w), but not in the unbalanced formula. The sample is accounted for in the application of Bayes' rule by looking at probabilities. A probability of a data point is given by taking into account the sample of the data. The code written to produce the previous Figures can be found in the 4.1 notebook.

2 Data 2 : Iris data

First, data had to be downloaded and imported using pandas from a csv file. The iris set is a 3-class 4D feature data set. The columns 0,1,2,3 represent the features of the data. The features are in the following order: sepal length, sepal width, petal length, petal width. A dataframe was created to include all the data with all the classes. A separation has been made in order to split the data into the three classes: 'Iris-setosa', 'Iris-versicolor' and 'Iris-virginica'. The data was then added to a nested array which contained three arrays, each array corresponding to each class. In order to solve the generalised eigenvalue condition for optimal weights, we had to compute the between-class and within-class variance-covariance matrices Σ_B and Σ_W. The between-class matrice has been calculated as follows: $\Sigma_B = \sum_c \frac{N_c}{N}(\mu_c - \mu)(\mu_c - \mu)^T$. μ stands for the mean of the class means and μ_c for the mean of the class. N_c stands for the number of points in class c, and N is the number of points in all three classes. In this case, $N_c = 50$, and $N = 150$. The within class matrix is calculated by adding the covariance matrices for each class. The between-class matrix was stored in S_B and the within-class matrix in S_W. In order to find the optimal direction w* for projecting the data onto, the generalised eigenvalue problem had to be solved $\Sigma_b w = \lambda \Sigma_w w$. The determinant of the within-class matrix is not zero, which means Σ_W is invertible. Because it is invertible, the generalized eigenvalue problem can be rearranged to an usual eigenvalue problem for the symmetric matrix $S_W^{-1} S_B : S_W^{-1} S_B v = \lambda v$. The eigenvalues and the eigenvector have been computed afterwards. In order to get the optimal w^* direction, the maximal eigenvalue and its corresponding eigenvector was selected. The vector was then normalised and the classes were projected onto the optimal direciton. Figure 5 illusrtates the separation between the classes when projected on the optimal w* direction. The values of the matrices, as well as the eigenvalues, eigenvectors and the optimal direction can be seen in the 4.2 notebook.

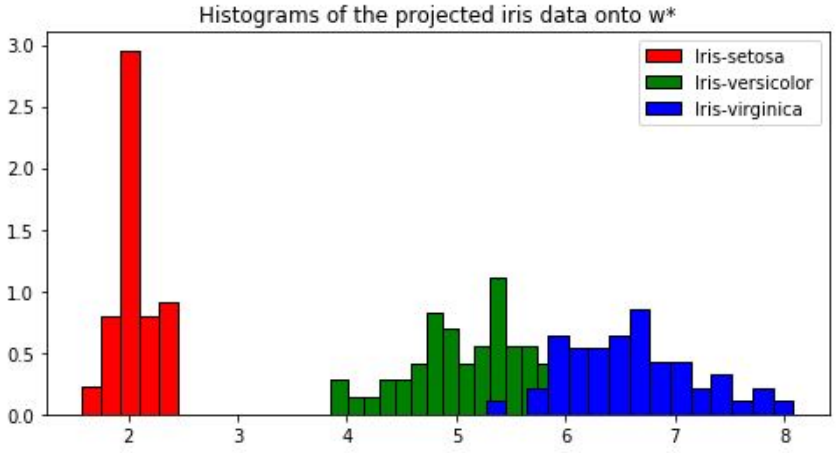


Figure 5: Projection onto the optimal direction w*

The histograms of the three classes in the reduced dimensional space defined by w^* clearly show the variance between the three classes. There is a small overlap in the iris-versicolor and iris-virginica, but this error rate is insignificant.

Figure 6 shows histogram of the three classes projected onto vector $w = w^* + a$. The amount of data loss when projecting on to this vector is substantial in comparison to the optimal eigenvector. There is a big overlap between iris-versicolor and iris-virginica, which makes it hard to see the variance between the data. The vector a was constructed out of the other generalised eigenvectors, which is another eigenvector onto which the data could have been transformed.

For the data generated by the two 2-dimensional Gaussian distributions, we have used Fisher's LDA to classify the data by using the log-odds presented in the calculations of the first section. The Fisher ratio was defined as follows:

$$F(w) \triangleq \frac{(\mu_a - \mu_b)^2}{\frac{n_a}{n_a+n_b}\sigma_a^2 + \frac{n_b}{n_a+n_b}\sigma_b^2} \tag{4}$$

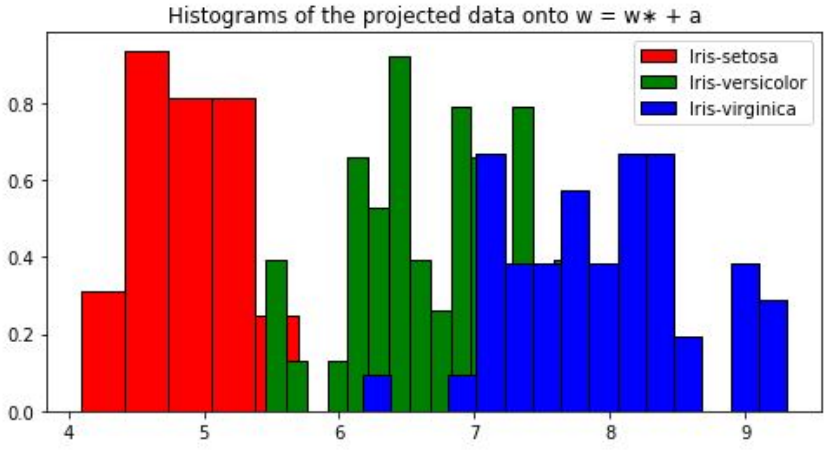


Figure 6: Projection onto $w = w^* + a$

In order to choose the optimal direction for vector w, one had to compute: $w^* = argmax_w J(w)$. This ratio can be rewritten to make the weight vector dependence explicit by using the between and within class scatter.

$$s_c^2 = \frac{1}{n_c} \sum_{x^n \in D_c} (y^n - m_c)^2 = \frac{1}{n_c} \sum_{x^n \in D_c} (w^T x^n - w^T \mu)^2 = \frac{1}{n_c} \sum_{x^n \in D_c} w^T (x^n - \mu)(x^n - \mu)^T w \triangleq w^T S w \tag{5}$$

Within class scatter:

$$n_+ s_+^2 + n_- s_-^2 = w^T (n_+ S_+ + n_- S_-) w \triangleq w^T S_W w \tag{6}$$

Between class scatter:

$$(m_+ - m_-)^2 = (w^T (\mu_+ - \mu_-))^2 = w^T (\mu_+ - \mu_-)(\mu_+ - \mu_-)^T w \triangleq w^T S_B w \tag{7}$$

$$J(w) = \frac{(m_+ - m_-)^2}{n_+ s_+^2 + n_- s_-^2} \rightarrow J(w) = \frac{w^T S_B w}{w^T S_W w} \tag{8}$$

In order to find the best direction, partial derivatives are set to 0:

$$\frac{\partial J(w)}{\partial w} = \frac{2}{(w^T S_W w)^2} ((w^T S_W w) S_B w - (w^T S_B w) S_W w) \tag{9}$$

This equation can be reduced to a generalised eigenvalue problem which can be extended to multiple classes, not just the 2 solved in the first part of the coursework.

$$S_B w = \lambda S_W w \tag{10}$$

By looking at the previously listed equations, one can notice how this was solved in order to find the optimal direction w* onto which to project the data. This is just another way of writing the Fisher formula. In the essence, they are both the same in the sense that they both try to achieve the same thing, which is finding the optimal vector. The eigenvalue problem is just a rewritten fisher ratio. The method presented above is used for a multi-class classification, while the other was used for a 2-class classification

3 Linear Regression with non-linear functions

For the first part of this exercise, one had to create a function $y(x) = \sin(x) + \epsilon$, where ϵ is a noise term. The number of points N were generated in the interval $0 \leq x \leq 2\pi$ and with the added noise value.

The weights $w = (w_0, ..., w_p)$ were learned by minimising the loss function using gradient descent:

$$\sum_{n=1}^{N}(y^n - w_0 - \sum_{j=1}^{p} w_j \phi_j(x^n))^2 + \lambda C(w) \tag{11}$$

The gradient for a loss function is:

$$(\bigtriangledown_w L)_i = \frac{\partial L}{\partial w_i} = -2\sum_{n=1}^{N} A_{ni}(y_n - \sum_{j=1}^{p} A_{nj} w_j) = -2(A^T(y - Aw))_i \tag{12}$$

The L_2 norm was used for the penalty of $C(w)$. This norm regulisation for minimising the squared residual with L_2 is called the ridge regression. :

$$C(w) = ||w||_2^2 = \bigtriangledown_w(||w||^2) = 2w \tag{13}$$

The produced data points from the sin function are displayed in Figure 7.

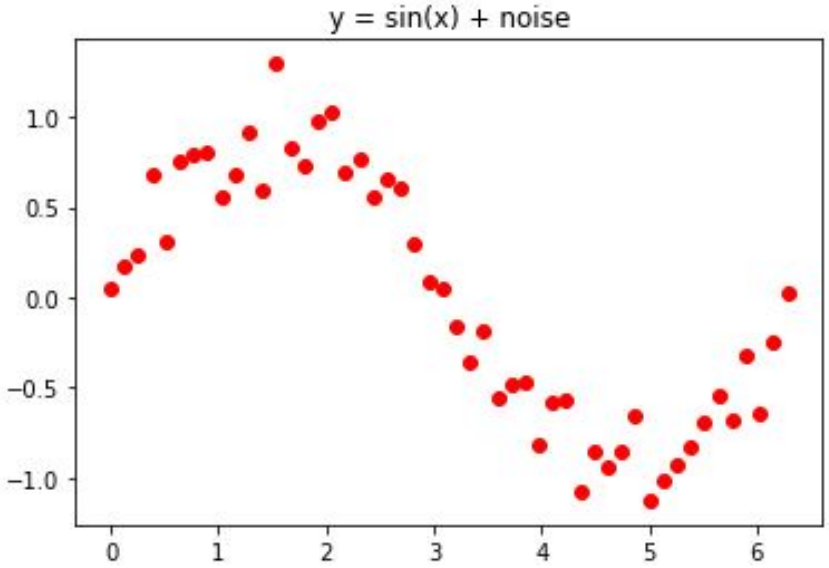
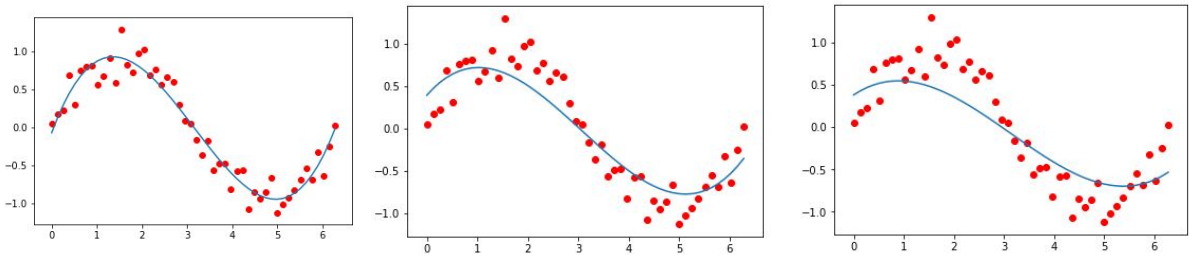


Figure 7: $y = \sin(x) + \epsilon$

For the L_2 norm penalty, the weights have been obtained from the analytical expression. II is an identity matrix with size (p + 1) x (p + 1):

$$w = (X^T X + \lambda \mathrm{II}_{p+1})^{-1} X^T y \tag{14}$$

Figure 8 shows the weights in relation to our N points. Different λ values have been chosen to illustrate how it influences the stretch of our weights. One can notice how a higher value of lambda shifts the polynomial function. Here, only positive values of lambda have been chosen, showing how our graph is being shifted by that value. A negative value would have showed a similar behaviour, but in a different direction.



(a) $\lambda = 0$ (b) $\lambda = 3$ (c) $\lambda = 9$

Figure 8: Plotted weight

Our initial set of points has been divided into training and test sets: S^{tr} and S^{ts}. Our training sets correspond to the color blue and our test sets correspond to the color red. The mean of the squared residuals was measured on the test set S^{ts} as a performance metric for each model. Figure 9 shows the degree of the polynomial used to fit our weights on the model. For this, a polynomial of degree 4 has been used.

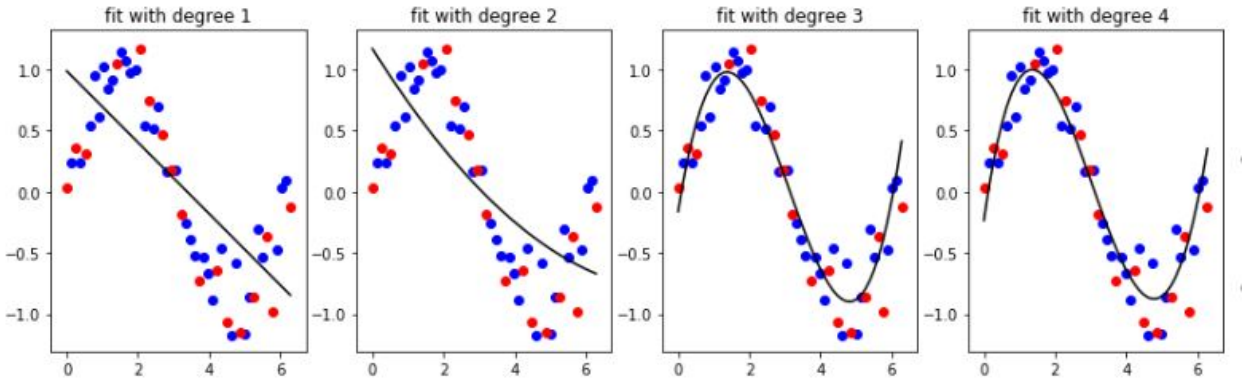


Figure 9: Polynomial functions fit to our data set

References

Hastie, T., Tibshirani, R. & Friedman, J. (2001), *The Elements of Statistical Learning*, Springer Series in Statistics, Springer New York Inc., New York, NY, USA.

Scikit-learn (n.d.), 'Linear and Quadratic Discriminant Analysis', https://scikit-learn.org/stable/modules/lda$_q$*da.html*. *Accessed* : $2018 - 11 - 25$.