Machine Learning 2017/2018
Home Assignment 1

Denis Trebula - (jmp640)

November 28, 2017

1 4. Probability Theory: Properties of Expectation

1.1 Section 1

I have decided to use mathematical way to simply prove following identity: $\mathbf{E}[X+Y] = \mathbf{E}[X] + \mathbf{E}[Y]$

$$\begin{aligned}
\mathbf{E}[X,Y] &= \sum_{x,y}(x+y)Prob(X=x,Y=y) && (1)\\
&= \sum_{x,y} xProb(X=x,Y=y) + \sum_{x,y} yProb(X=x,Y=y) && (2)\\
&= \sum_{x} x\sum_{y} Prob(X=x,Y=y) + \sum_{y} y\sum_{x} Prob(X=x,Y=y) && (3)\\
&= \sum_{x} xProb(X=x) + \sum_{y} yProb(Y=y) \quad = \mathbf{E}[X] + \mathbf{E}[Y] && (4)
\end{aligned}$$

Trough steps 3 and 4 I have used the rule of total probability, which is basically: $\sum_{x} Prob(X=x, Y=y) = Prob(X=x)$

1.2 Section 2

Proof of: $\mathbf{E}[XY] = \mathbf{E}[X]\mathbf{E}[Y]$

$$\begin{aligned}
\mathbf{E}[XY] &= \sum_{x,y} Prob(X=x,Y=y) && (5)\\
&= \sum_{x}\sum_{y} xyProb(X=x,Y=y) && (6)\\
&= \sum_{x}\sum_{y} xyProb(X=x)Pr(Y=y) && (7)\\
&= \sum_{x} xProb(X=x)\sum_{y} yProb(Y=y) = \mathbf{E}[X]\mathbf{E}[Y] && (8)
\end{aligned}$$

At step 6 I follow the assumption that X and Y are independent random variables. Based on which we could continue with proof.

1.3 Section 3

n	0	1
0	1/4	2/4
1	1/4	1/4

$$\begin{aligned}
\mathbf{E}[X] &= \left(1\times\frac{2}{4}\right) + \left(0\times\frac{3}{4}\right) = \frac{2}{4}\\
\mathbf{E}[Y] &= \left(1\times\frac{3}{4}\right) + \left(0\times\frac{2}{4}\right) = \frac{3}{4}\\
\mathbf{E}[XY] &= \left(1\times1\times\frac{1}{4}\right) + \left(1\times0\times\frac{1}{4}\right) + \left(0\times1\times\frac{2}{4}\right) + \left(0\times0\times\frac{1}{4}\right)\\
&= \frac{1}{4}
\end{aligned}$$

We can see that therefore: $\mathbf{E}[X]\times\mathbf{E}[Y] = \frac{1}{2}$ it is clearly obvious that $\mathbf{E}[X]\times\mathbf{E}[Y] \neq \mathbf{E}[XY]$

1.4 Section 4

Goal is to prove that:

$$\mathbb{E}[\mathbb{E}[X]] = \mathbb{E}[X] \tag{9}$$

Let $C = \mathbb{E}[X]$. Then

$$\mathbb{E}[\mathbb{E}[X]] = \mathbb{E}[C] = \sum_{c \in \text{Ran}(C)} c \cdot p_C(c) = \sum_{c \in \text{Ran}(C)} c \cdot \mathbb{P}_C(C = c) = c \cdot 1 = C = \mathbb{E}[X]$$

We can remove the sum since $\mathbb{P}_C(C = c) = 1$ as C is just a constant.

1.5 Section 5

We want to prove that : $\mathbf{E}\left[(X - \mathbf{E}[X])^2\right] = \mathbf{E}[X^2] - (\mathbf{E}[X])^2$

$$\begin{aligned}
\mathbf{E}\left[(X - \mathbf{E}[X])^2\right] &= \mathbf{E}\left[X^2 - 2\mathbf{E}[X]X + (\mathbf{E}[X])^2\right] \\
&= \mathbf{E}[X^2] - 2\mathbf{E}[X]\mathbf{E}[X] + (\mathbf{E}[X])^2 \\
&= \mathbf{E}[X^2] - 2(\mathbf{E}[X])^2 + (\mathbf{E}[X])^2 = \mathbf{E}[X^2] - (\mathbf{E}[X])^2
\end{aligned}$$

Therefore we have proved that: $\mathbf{E}\left[(X - \mathbf{E}[X])^2\right] = \mathbf{E}[X^2] - (\mathbf{E}[X])^2$.

2 5. Probability Theory: Complements of Events

2.1 Section 1

First we need to denote the complement of an event A by $\overline{A}$ as required. We then determine the complement as $\overline{A} = \Omega \diagup A$, therefore nothing that A and its complement are mutually exclusive, which means we can write $Pr[A \cup B] = Pr[A] + Pr[B]$.
Based on that we can write following proof of $Pr[A] = 1 - Pr[\overline{A}]$:

$$\begin{aligned}
1 = Pr[S] &= Pr[A \cup \overline{A}] = Pr[A] + Pr[\overline{A}] \\
1 &= Pr[A] + Pr[\overline{A}] \\
Pr[A] &= 1 - Pr[\overline{A}]
\end{aligned}$$

Therefore coming to conclusion that $Pr[A] = 1 - Pr[\overline{A}]$.

2.2 Section 2

2.2.1 a

We are flipping coin 10 times, we can assume that there can be result in only one outcome where there is not atleast one tail. In other words, the outcome will be all heads. The probability of getting this outcome is:

$$P(\textit{All heads}) = \frac{1}{2^{10}} = \frac{1}{1024}$$

Now we can say that from the assumptions above we know that getting all other outcomes, in this case all with at least one tail, is equal to

$$P(\textit{At least one tail}) = 1 - \frac{1}{1024} = \frac{1023}{1024}$$

that is approximately 0.999 which means that probability of getting at least one tail is large.

2.2.2 b

To find the probability of observing at least two tails we can use the binomial distribution which is calculated below.

$$Pr[n, k] = \binom{n}{k} p^k (1 - p)^{n-k}$$

The binomial distribution can be calculated, by taking the complement of the event that either 1 or 0 tails are evident. This reflects in following equation:

$$Pr[A] = 1 - \left(\binom{10}{1} \left(\frac{1}{2}\right)^1 \left(1 - \frac{1}{2}\right)^9 + \binom{10}{0} \left(\frac{1}{2}\right)^0 \left(1 - \frac{1}{2}\right)^{10} \right) = \frac{1013}{1024}$$

Again we found out that the probability of observing at least two tails is high.

3 6. Digits Classification with Nearest Neighbors

In this question I applied nearest neighbors learning algorithm to classify handwritten digits. Python as programming language was used to compute and depict the results, to plot graphs I used matplotlib which is well known in python community. Others libraries such as numpy or pandas were used as well. The algorithm is implemented in python script Assigment1.py.

In the plots are depicted comparisons between digits 0 and 1, 0 and 8, 5 and 6, on the X axis is number of neighbors and on the Y axis is the percentage of errors. If we used simple the number of errors divided by the number of tries, the plot will look different. Percentage is in my case more preferable.

First we load all the MNIST files. We reshape mentioned files and then we follow with calculations and evaluations. We tested the algorithm on classifying zeros from ones, zeros from eights and fives from sixes. We tested validation data against the training data as requested in Assignment thus first 80 percent of data against the last 20 percent and then with the test data against the entire set of the training data.

3.1 Plot 1

In the plot are depicted comparisons between digits 0 and 1.
0 errors were found due to difference between the numbers 0 and 1. They share almost none coordinates in a bitmap.

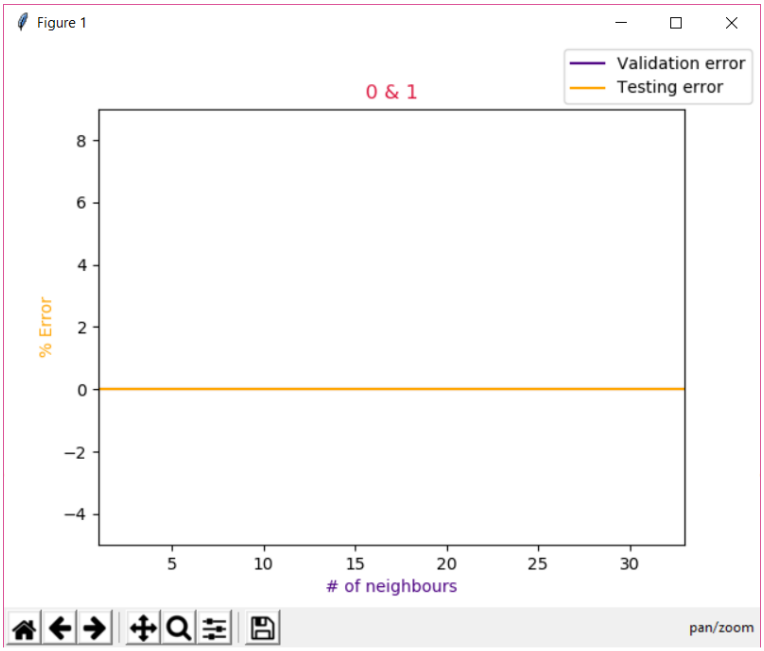

Figure 1: Data plotted as lines

3.2 Plot 2

In the plot are depicted comparisons between digits 0 and 8.
We were able to classify the data more accurately.

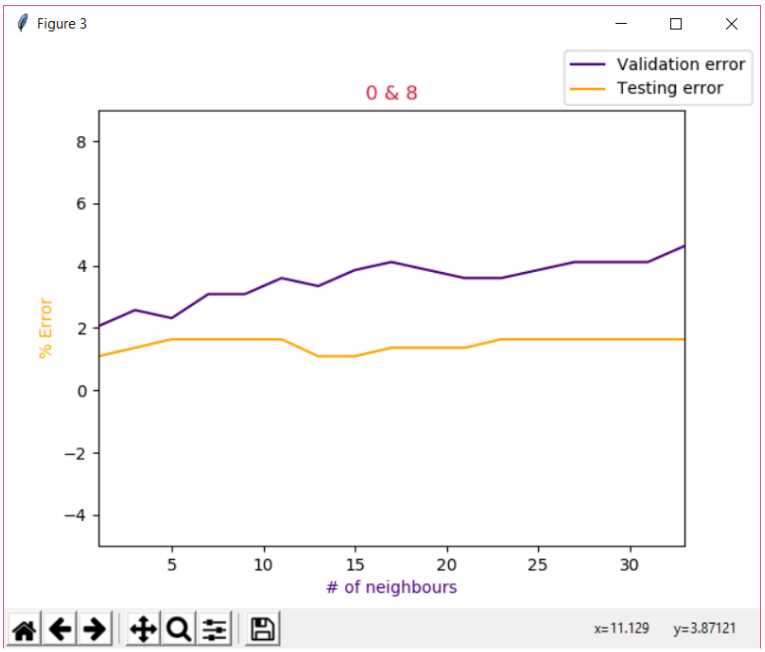

Figure 2: Data plotted as lines

3.3 Plot 3

In the plot are depicted comparisons between digits 5 and 6.
The classification regarding 5s and 6s were really similar which results in having very close values in the whole diagram. The optimum values for both the validation data and the training data were the same, showing that it will be relatively hard to increase or improve on using a 3 Nearest neighbors algorithm

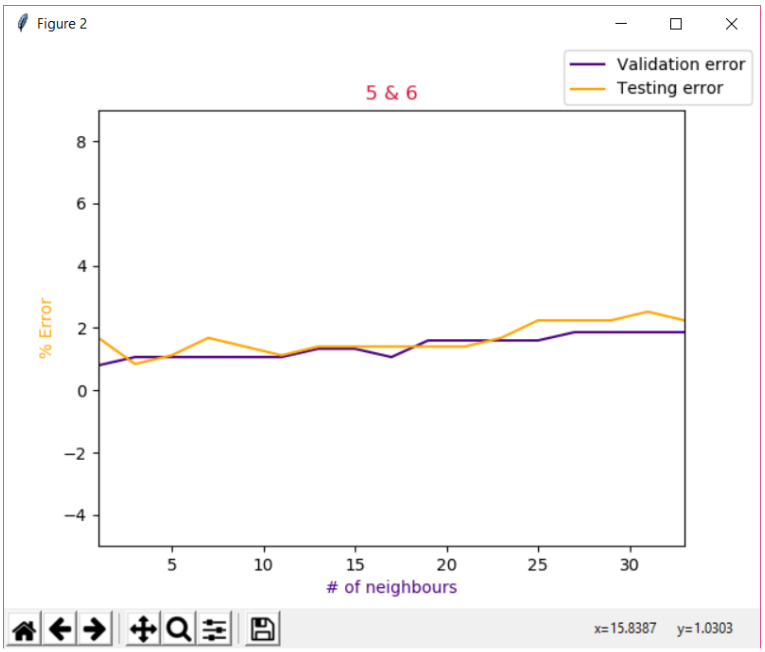

Figure 3: Data plotted as lines

4 7. Linear regression

Implemented algorithm for linear regression can be found in file Assigment1.py. We used competent python mathematical libraries to get the right results e.g. matplotlib, numpy, pandas etc.

4.1 Question 5.1

4.1.1 1

First the data is loaded from the file `DanWood.dt`, the data is then prepared and calculations follow.

4.1.2 2

Then we retrieve results from running, therefore we get two parameters

```
Parameteres:

[ 1.309  2.138]
[ 1.471  3.421]
[ 1.49   3.597]
[ 1.565  4.34 ]
[ 1.611  4.882]
[ 1.68   5.66 ]

Parameters: weight = [ -10.42696146 9.48934569]

Mean Squared Error = [0.0124342216151]
```

where $w \approx 9.49$ and $b \approx -10.43$.

The mean squared error is defined as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

The RMSE of the linear model performed on the data is 0.0124

4.1.3 3

The plot is obtained by using the library matplotlib which depicts figure below. The X axis shows the absolute temperature and Y axis depicts Radiated energy. We can clearly see the data points and regression line.

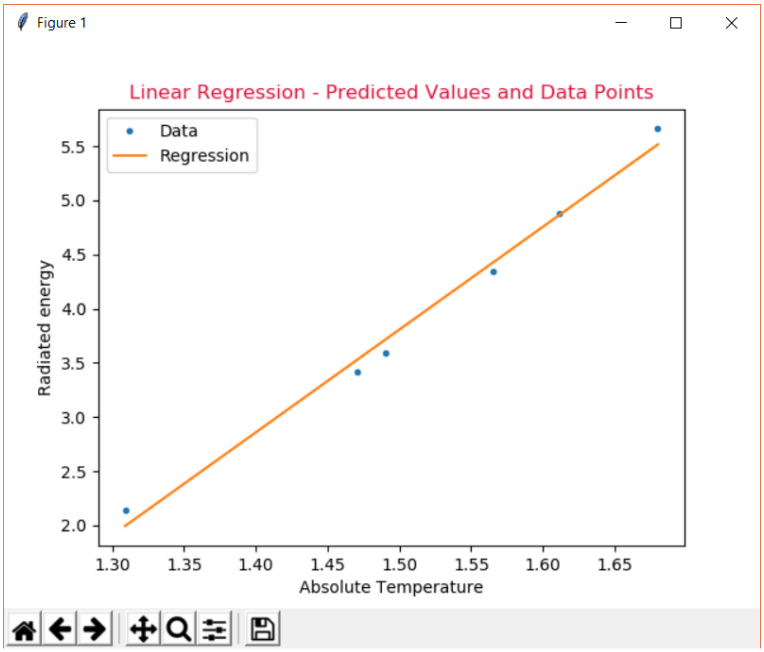

Figure 4: Data plotted as points, the linear regression model plotted as a line

4.1.4 4

To calculate variance we need second column from DanWood.dt or in other words column y. to which we apply following equation:

Variance $= \frac{\sum(y-\mu)^2}{6} = 1.26$

The mean squared error for all n x values resulted in 0.0124. If we divide mean squared error with our variance, we get really small result which means that our linear regression model is relatively competitive.