ELEN E4903 Homework 2

Kliment Mamykin UNI 2770

Problem 1 a: Derive $\hat{\pi}$ for the prior

$$\begin{aligned}
\hat{\pi}_{ml} &= \arg\max_{\pi} \sum_{i=1}^{n} \ln p(y_i \mid \pi) \\
&= \arg\max_{\pi} \sum_{i=1}^{n} \ln(\pi^{y_i}(1-\pi)^{1-y_i}) \\
&= \arg\max_{\pi} \sum_{i=1}^{n} y_i \ln \pi + (1-y_i)\ln(1-\pi) \\
&= \arg\max_{\pi} \sum_{i=1}^{n} y_i \ln \pi + \sum_{i=1}^{n}(1-y_i)\ln(1-\pi)
\end{aligned}$$

$$\Downarrow$$

$$\begin{aligned}
0 &= \frac{d}{d\pi}\left(\sum_{i=1}^{n} y_i \ln \pi + \sum_{i=1}^{n}(1-y_i)\ln(1-\pi)\right)\Big|_{\hat{\pi}_{ml}} \\
&= \frac{\sum_{i=1}^{n} y_i}{\hat{\pi}_{ml}} - \frac{\sum_{i=1}^{n}(1-y_i)}{1-\hat{\pi}_{ml}}
\end{aligned}$$

$$\Downarrow$$

$$\hat{\pi}_{ml} = \frac{\sum_{i=1}^{n} y_i}{n}$$

Problem 1 b: Derive $\hat{\theta}_y^{(1)}$ for Bernoulli features

Objective:

$$\hat{\theta}^{(1)}_{y\,ml} = \arg\max_{\theta_y^{(1)}} \sum_{i=1}^{n} \ln p(x_{i\,1} \mid \theta^{(1)}_{y_i}) \quad y \in \{0, 1\}$$

To simplify notation lets temporarily disregard the dimention/feature index, and define $\hat{\theta}_{y\,ml} = \hat{\theta}^{(1)}_{y\,ml},\ \theta_y = \theta_y^{(1)},\ x_i = x_{i\,1}$, and consider that $\hat{\theta}_{y\,ml}$ is a vector of length 2 for each value of y , containing class conditional parameters for a Beroulli :

$$\begin{bmatrix} \hat{\theta}_{0\,ml} \\ \hat{\theta}_{1\,ml} \end{bmatrix} = \begin{bmatrix} \arg\max_{\theta_0} \sum_{i=1}^n \mathbb{1}(y_i = 0) \ln p(x_i \mid \theta_0) \\ \arg\max_{\theta_1} \sum_{i=1}^n \mathbb{1}(y_i = 1) \ln p(x_i \mid \theta_1) \end{bmatrix} = \begin{bmatrix} \frac{\sum_{i=1}^n x_i 1(y_i=0)}{\sum_{i=1}^n 1(y_i=0)} \\ \frac{\sum_{i=1}^n x_i 1(y_i=1)}{\sum_{i=1}^n 1(y_i=1)} \end{bmatrix}$$

Collapsing the 0 and 1 indices into y

$$\hat{\theta}_{y\,ml} = \frac{\sum_{i=1}^n x_i \mathbb{1}(y_i = y)}{\sum_{i=1}^n \mathbb{1}(y_i = y)} \quad y \in \{0, 1\}$$

And bringing the original notation for the first dimension:

$$\hat{\theta}^{(1)}_{y\,ml} = \frac{\sum_{i=1}^n x_{i\,1} \mathbb{1}(y_i = y)}{\sum_{i=1}^n \mathbb{1}(y_i = y)} \quad y \in \{0, 1\}$$

Problem 1 c: Derive $\hat{\theta}_y^{(2)}$ for Pareto features

Objective:

$$\hat{\theta}^{(2)}_{y\,ml} = \arg\max_{\theta_y^{(2)}} \sum_{i=1}^n \ln p(x_{i\,2} \mid \theta^{(2)}_{y_i}) \quad p(x \mid \theta) = \theta x^{-\theta+1} \quad y \in \{0, 1\}$$

First we derive the maximum likelihood estimator for θ in a Pareto distributed sample.

Let $X = (x_0, x_1, \dots, x_n)$ be i.i.d. r.v. $X \sim Pareto(\theta) \Rightarrow p(x_i \mid \theta) = \theta x_i^{-\theta+1}$

$$L(\theta) = p(X \mid \theta) = \prod_{i=1}^n p(x_i \mid \theta)$$

$$\ell(\theta) = \ln L(\theta) = \sum_{i=1}^n \ln p(x_i \mid \theta)$$

$$= \sum_{i=1}^n \ln \theta x_i^{-\theta+1}$$

$$= n \ln \theta - (\theta + 1) \sum_{i=1}^n \ln x_i$$

$$\frac{d\ell(\theta)}{d\theta} = \frac{n}{\theta} - \sum_{i=1}^n \ln x_i = 0$$

$$\hat{\theta}_{ml} = \frac{n}{\sum_{i=1}^n \ln x_i}$$

Now back to class conditional probabilities:

$$\hat{\theta}^{(2)}_{y\,ml} = \arg\max_{\theta^{(2)}_y} \sum_{i=1}^{n} \ln p(x_{i\,2} \mid \theta^{(2)}_{y_i})$$

$$\hat{\theta}^{(2)}_{0\,ml}, \hat{\theta}^{(2)}_{1\,ml} = \arg\max_{\theta^{(2)}_0, \theta^{(2)}_1} \sum_{i=1}^{n} \mathbb{1}(y_i = 0) \ln p(x_{i\,2} \mid \theta^{(2)}_0) + \sum_{i=1}^{n} \mathbb{1}(y_i = 1) \ln p(x_{i\,2} \mid \theta^{(2)}_1)$$

$$= \frac{\sum_{i=1}^{n} \mathbb{1}(y_i = 0)}{\sum_{i=1}^{n} \mathbb{1}(y_i = 0) \ln x_i}, \frac{\sum_{i=1}^{n} \mathbb{1}(y_i = 1)}{\sum_{i=1}^{n} \mathbb{1}(y_i = 1) \ln x_i}$$

$$\hat{\theta}^{(2)}_{y\,ml} = \frac{\sum_{i=1}^{n} \mathbb{1}(y_i = y)}{\sum_{i=1}^{n} \mathbb{1}(y_i = y) \ln x_i}$$

Problem 2 a: Naive Bayes and confusion matrix

In [15]:
```
%load_ext autoreload
%autoreload 2
%matplotlib inline
from hw2 import hw2
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

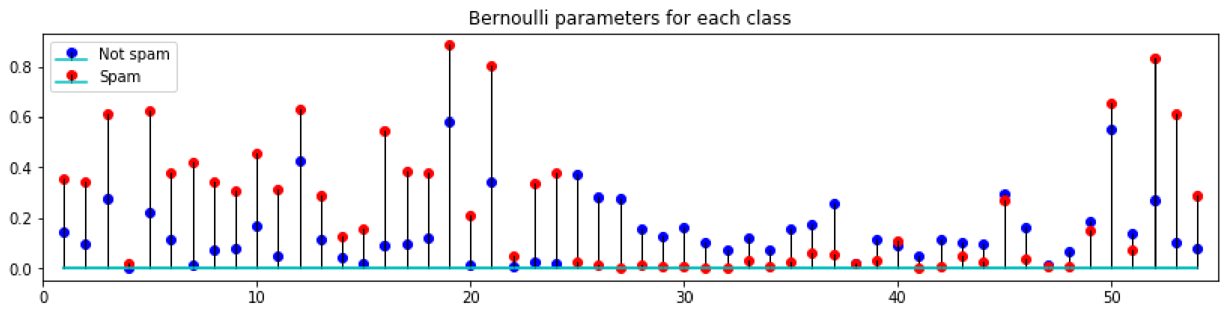
In [16]:
```
hw2.problem_2_part_a()
```

	Predicted 0	**Predicted 1**
True 0	54	2
True 1	5	32

```
Accuracy: 92.47
```

Problem 2 b: Bernoulli parameters visualization

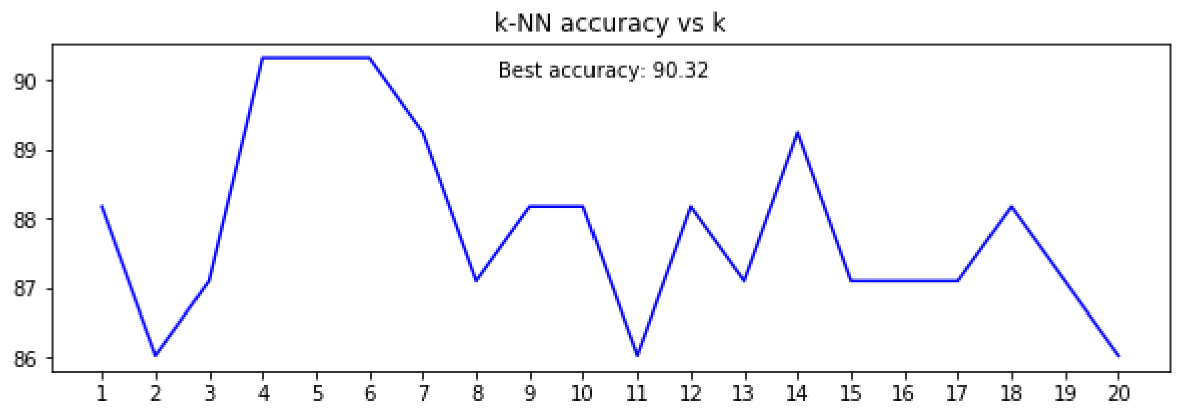
In [17]:
```
hw2.problem_2_part_b()
```



The dimention 16 represent the presence of the word "free" and dimension 52 represents the presence of a character "!". For both of these dimensions the parameters for the "spam" class are higher then the parameter for "not spam" class, hinting that the presence of these words/chars are more likely in "spam" emails. The value of the parameter represent relative frequency when this word appeared in "spam" vs "not spam" emails. Concretely, the word "free" appeared in about 60% of "spam" emails, and 10% of "not spam" emails. The "!" character appeared in ~85% of "spam" emails and ~25% of "not spam" emails.

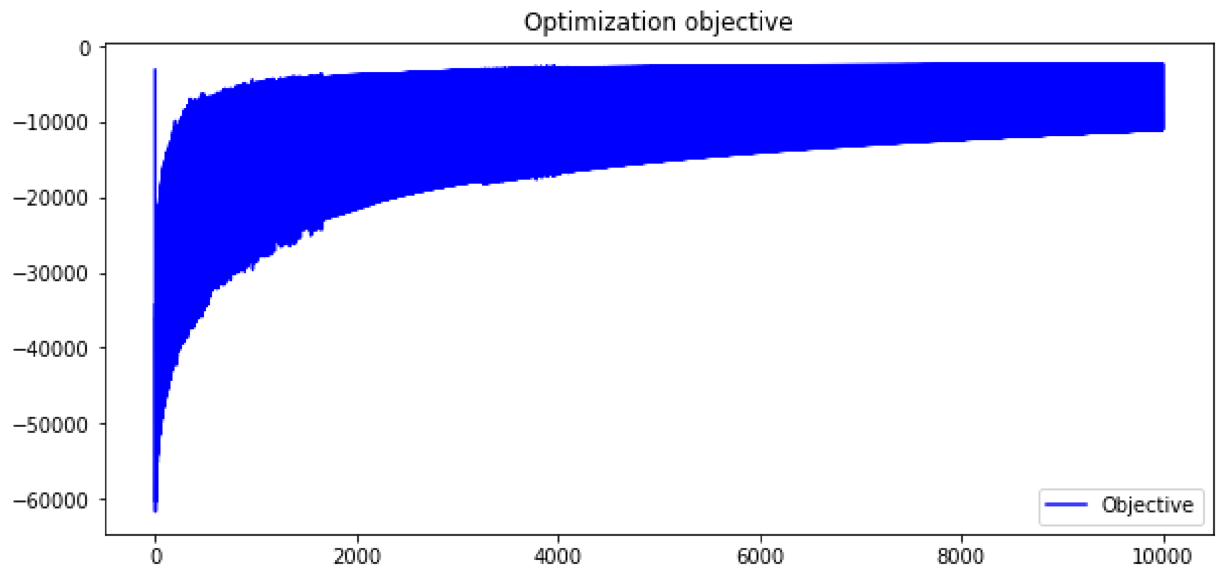
Problem 2 c: k-NN accuracy plot

In [18]:
```
hw2.problem_2_part_c()
```



Problem 2 d: Logistic Regression steepest ascend

In [23]:
```
hw2.problem_2_part_d()
```

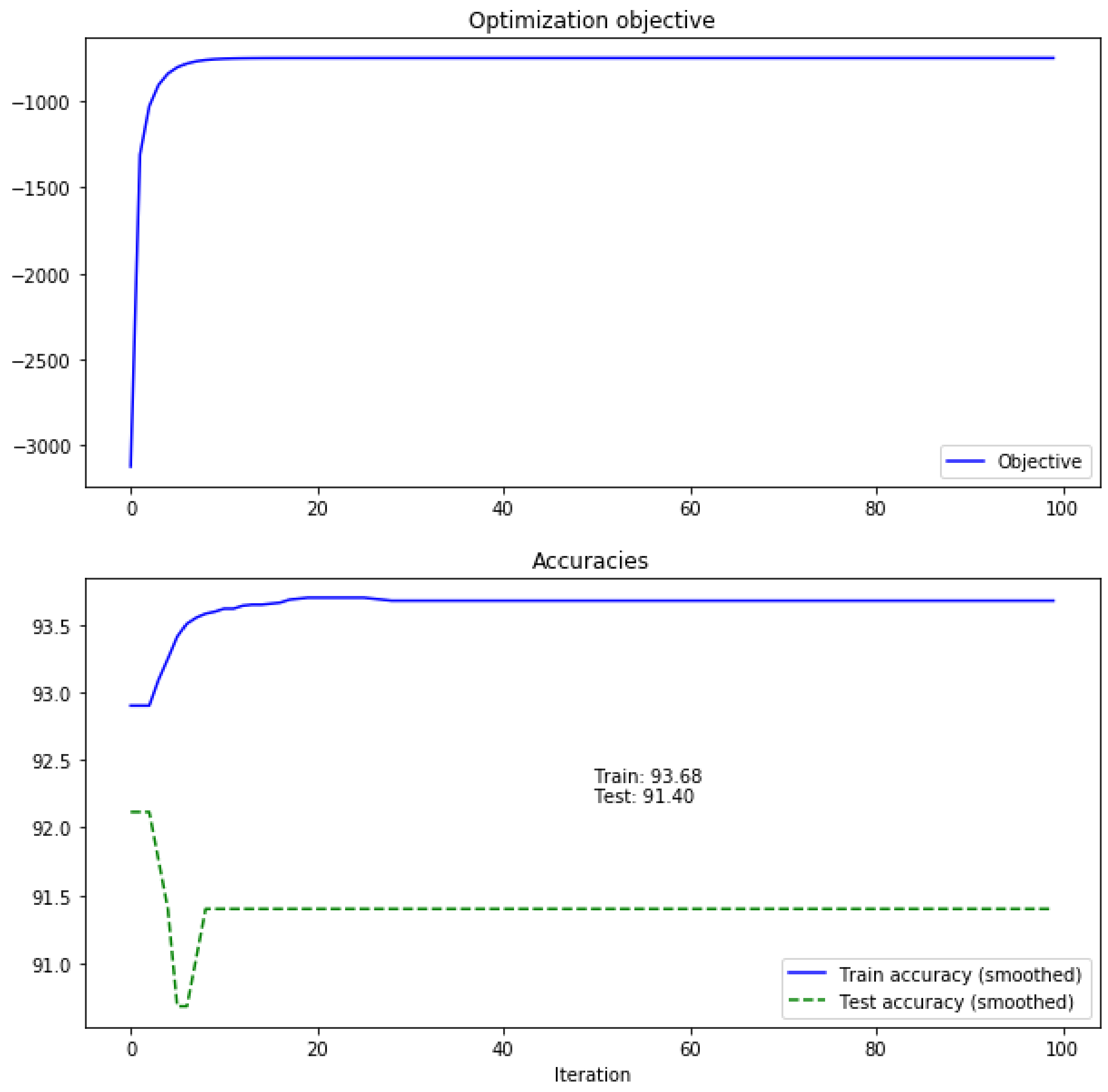


```
Test accuracy: 68.82
```

The objective function oscillates and is very noisy. So are the accuracies on training and test data (not shown). This indicates a very narrow canyon in the surface of the objective function, and the learning is too large to "stop inside" of this canyon, it constantly overshoots a good location. This is probably caused by non-normalized features (e.g. Pareto features are much larger). However, the algorythm converges (just very slow). I ran it for 2e5 iterations and the objective still increases and the accuracies improve.

Problem 2 e: Logistic Regression with Newton method

In [24]:
```
hw2.problem_2_part_e()
```



```
Test accuracy: 91.40
```