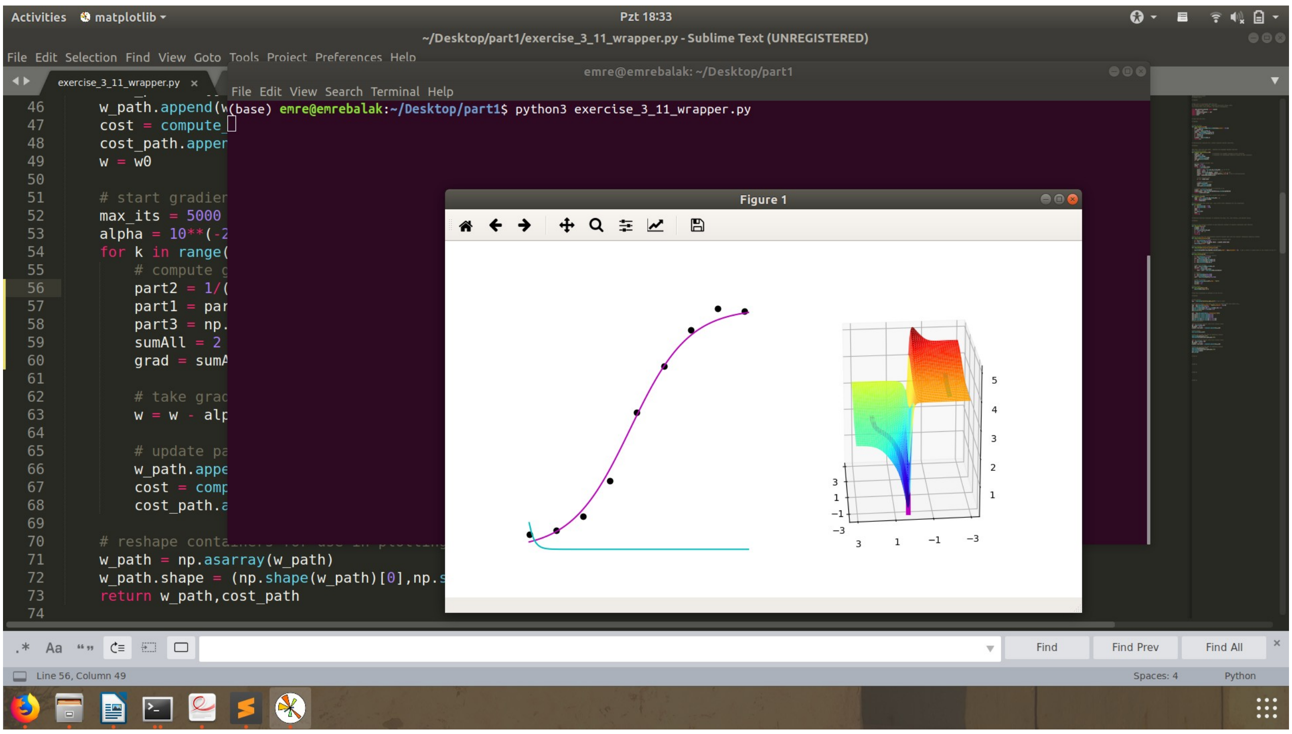
Yunus Emre BALAK – 150160509 13.05.2019
BLG 454 – HW 2

## PART 1) Code up gradient descent for logistic regression

### Table 1 – Code

| The added/changed code statement | Explanation |
|---|---|
| part2 = 1/(1 + my_exp(-np.dot(X,w))) | Implementation of this formula: $\sigma(\widetilde{x}_p^T \widetilde{w})$ |
| part1 = part2 - y | Implementation of this formula: $\sigma(\widetilde{x}_p^T \widetilde{w} - y_p)$ |
| part3 = np.ones(y.shape) - part2 | Implementation of this formula: $1-\sigma(\widetilde{x}_p^T \widetilde{w})$ |
| sumAll = 2 * np.sum(part1*part2*part3*X, axis=0) | Sum of multiplication all parts including X |
| grad = sumAll[np.newaxis,:].T | $\sum_{p=1}^{P} \sigma(\widetilde{x}_p^T \widetilde{w} - y_p \sigma(\widetilde{x}_p^T \widetilde{w})(1-\sigma(\widetilde{x}_p^T \widetilde{w})))$ This is the gradient formula. Operations divided in parts to get this formula. It creates a new dimension and transpoze of it. |

### Screen shot of the solution



Yunus Emre BALAK – 150160509 13.05.2019
BLG 454 – HW 2

**PART 2) Code up gradient descent for l2 regularized logistic regression**

**Table 2 – Code**

| The added/changed code statement | Explanation |
|---|---|
| part2 = 1/(1 + my_exp(-np.dot(X,w))) | Implementation of this formula: $\sigma(\tilde{x}_p^T \tilde{w})$ |
| part1 = part2 - y | Implementation of this formula: $\sigma(\tilde{x}_p^T \tilde{w} - y_p)$ |
| part3 = np.ones(y.shape) - part2 | Implementation of this formula: $1-\sigma(\tilde{x}_p^T \tilde{w})$ |
| sumAll = 2 * np.sum(part1*part2*part3*X, axis=0) | Sum of multiplication all parts including X to get the formula |
| w_regularization[:] = w[:] | It equals w_reg to entire w |
| w_regularization[0] = 0 | It makes zero the bias. |
| grad = sumAll[np.newaxis,:].T + 2*lam*w_regularization | $\sum_{p=1}^{P} \sigma(\tilde{x}_p^T \tilde{w} - y_p \sigma(\tilde{x}_p^T \tilde{w})(1-\sigma(\tilde{x}_p^T \tilde{w})))$ This is the gradient formula. Operations divided in parts to get this formula. And plus this formula $2\lambda\begin{bmatrix}0\\w\end{bmatrix}$ This normalises it. |

**Screen shot of the solution**