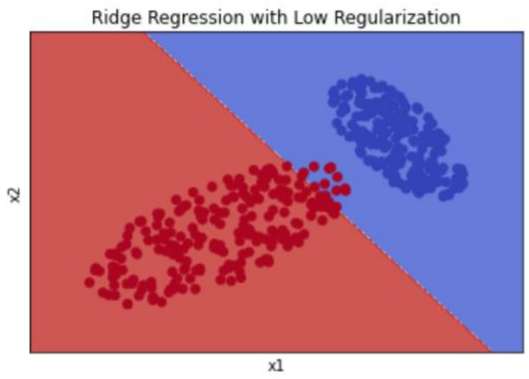
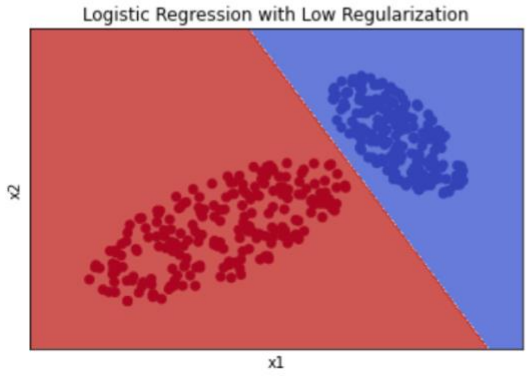
1. Comparing Different Loss Functions

a. Squared loss, commonly used for regression, is a poor choice for classification because it does not consider the discrete output nature and can attribute high loss to correctly classified points far from the line/hyperplane while attributing low loss to misclassified points close to it, leading to overfitting and suboptimal performance. Cross-entropy (Log) loss, on the other hand, is more appropriate for classification as it minimizes the difference between predicted class probabilities and the true class label and is less prone to overfitting.

b.

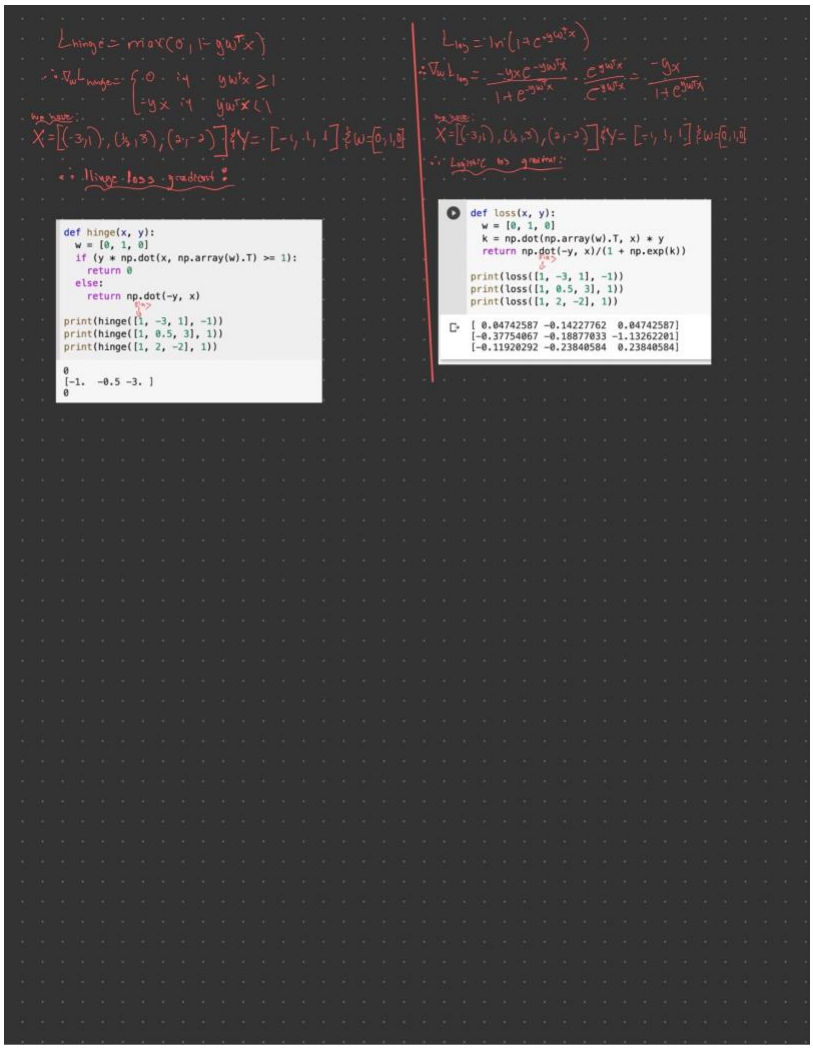
```
ridge = Ridge(alpha = 1e-6)
ridge.fit(X, Y)
make_plot(X, Y, ridge, "Ridge Regression with Low Regularization", "Ridge.png")
logistic = LogisticRegression(C = np.Infinity)
logistic.fit(X, Y)
make_plot(X, Y, logistic, "Logistic Regression with Low Regularization", "Logistic.png")
```





(See Code) The logistic regression boundary line is a better separator of the red and blue data points then the ridge regression boundary line which does not even completely separate the two as we still have misclassified red points. This is a result of squared loss, used by ridge regression, penalizing correctly classified points that are further from the boundary line hence shifting the boundary line towards those points to minimize loss and at the red data points have more points further from the boundary line hence the line is shifted to towards the red data points to minimize loss, however, it misclassified points at

the same time. However, log loss penalizes misclassified values and produces a lower loss for correctly classified points further from the boundary.

c. 

The loss of for each point corresponds to the printed value for that input x in each function call starting after x[0] which is the bias.

d. The hinge loss gradient converges to zero when all points are correctly classified meaning y(transpose(w))x >= 1 which results in 0. The log loss gradient converges to zero when y(transpose(w))x goes to infinity which makes the equation from part c zero. For linearly separable data, training error can be minimized by scaling the weights for correctly classified points hence allowing both losses to converge to zero. However, this will not change the decision boundary, but it will minimize the loss from both functions as it will increase y(transpose(w))x hence making it greater than or equal to 1 for hinge and closer to infinity for log loss hence reducing both losses.

e. Minimizing Lhinge loss alone does not change the decision boundary and only focuses on minimizing the loss. However, by adding a term that constrains the magnitude of the weight vector, the optimization problem becomes one of minimizing the loss by maximizing the weight margin while still correctly classifying the data points. This

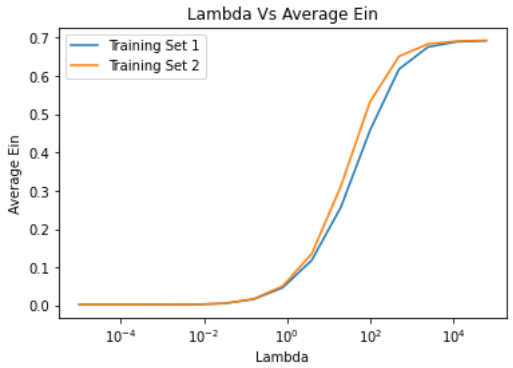
ensures that the decision boundary is not only optimal in terms of minimizing loss but also in terms of correctly classifying the data points.

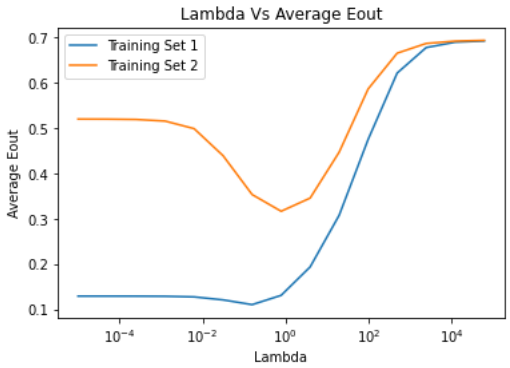
2. Effects Of Regularization

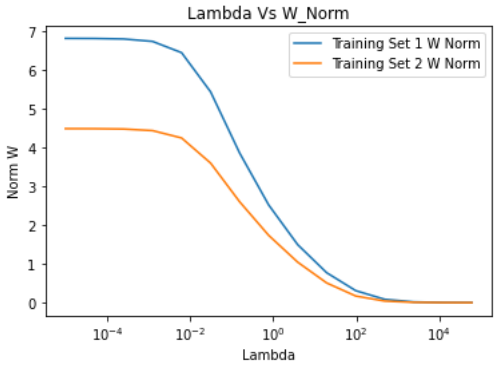
a. Adding the regularization term will not decrease the in-sample (training) error and it might actually increase the in-sample error (or keep it the same) because it is used to avoid overfitting by adding a constraint to the model to achieve simpler models that do not fit noise hence it cannot decrease the in-sample error and might actually increase it as a result of potential underfitting. Adding the regularization term might decrease the out of sample (test) error if there was overfitting prior to adding regularization penalty as the addition will prevent this overfitting. However, if the regularization term is too large, then it might actually increase the out of sample error as our model will be underfitting the data hence making it a bad generalizer.

b. L0 regularization is rarely used because it is not continuous and not convex hence not differentiable therefore making it extremely difficult to optimize using our gradient methods.

c.







(See Code)

d. Looking at the testing error we see that the wine training set with 100 points had a lower average Eout (out of sample error) for low lambda before increasing and starting to converge with the Eout of the dataset with 40 points for higher lambda towards the end. However, they had similar (minute differences in) Ein (Training Error) despite training set 1 having more training points. For training error, we see that for higher lambda values the training set with less points increases at a faster rate because it is smaller hence as the model gets more regularized, the training error for this set is more likely to to change due to limits in model complexity.

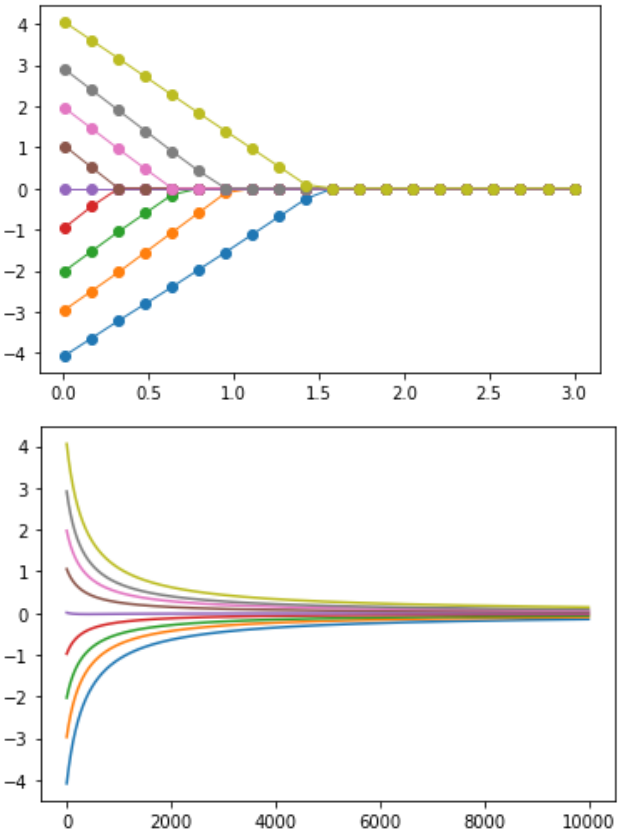
e. The training error (Ein) for wine_training1.txt increases with higher lambda values hence causing underfitting as lambda increases. The test test error (Eout) is lower with no regularization and lower values for lambda implying no overfitting with the training set until around lambda equals 1 where i began to increase hence showing underfitting for higher lambda values.

f. The L2 norm of w for wine_training1.txt decreases as lambda increases as the model is less constrained by the regularization term due to the equivalence between the norm constraint and regularization with the lagrangian.

g. If the model were trained with wine training2.txt, I would choose lambda = 0.00001 * (5^7) which is the point where Eout is the lowest hence allowing the model to best generalize.

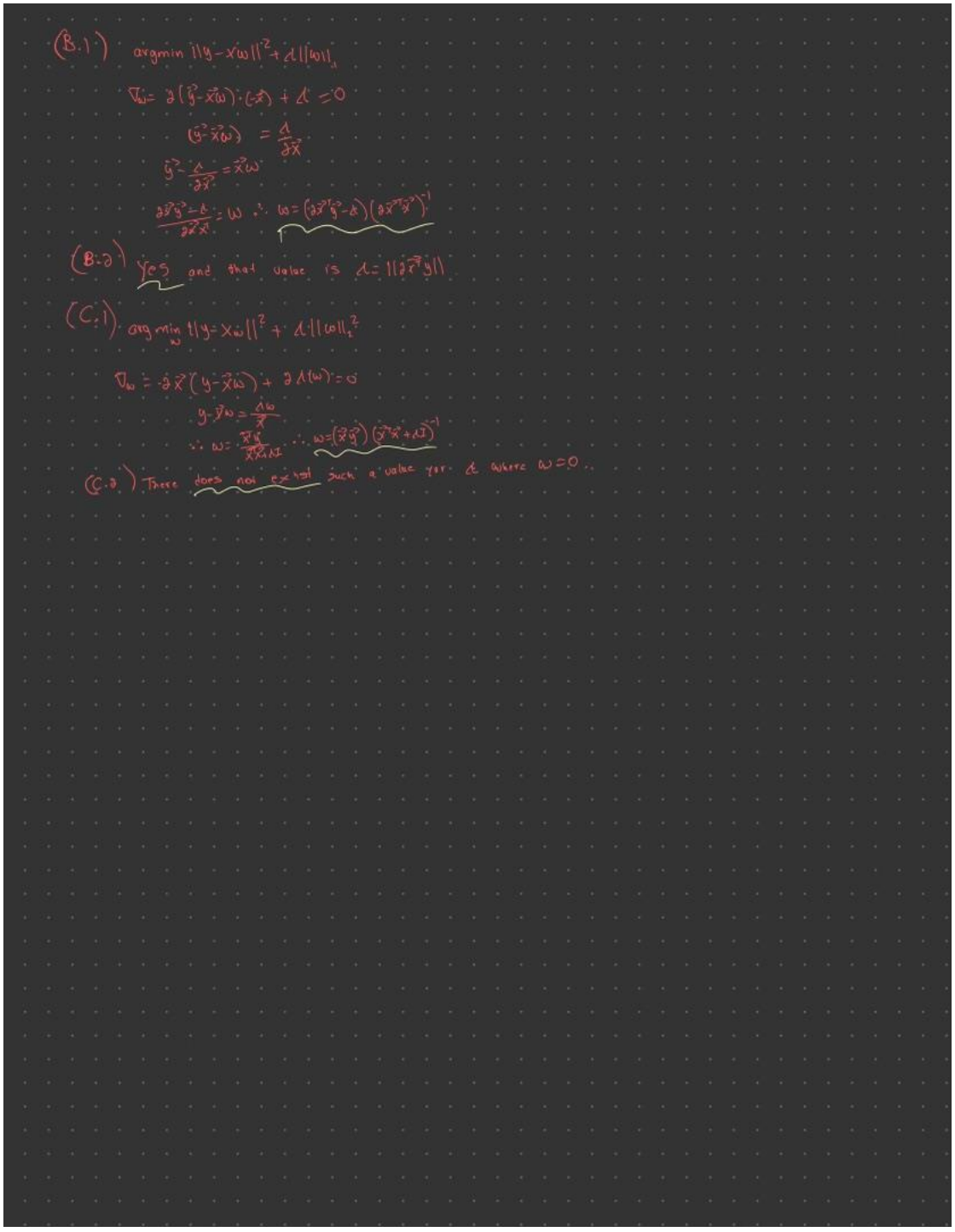
3. **Lasso (l_1) vs. Ridge (l_2) Regularization**

a.



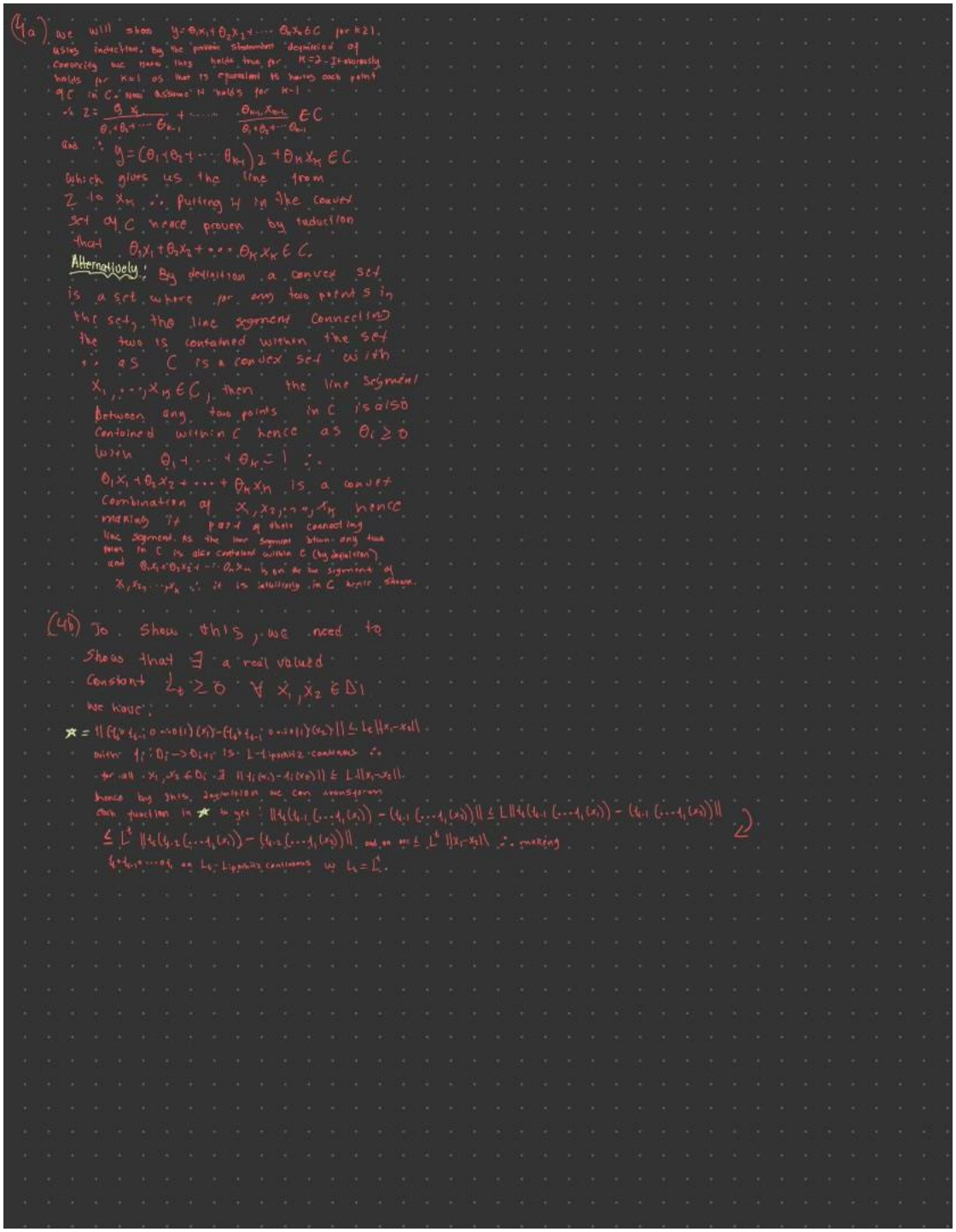
(See Code) As the regularization parameter increases, the number of model weights that are exactly zero increases with Lasso regression. For Ridge regression asymptotically approaches zero as the regularization parameter increases without ever reaching zero.

b. (B and C)



4. Convexity and Lipschitz Continuity

a. (a and b).


(4a) We will show $y=\theta_1x_1+\theta_2x_2+\cdots+\theta_kx_k \in C$ for $k \geq 1$.
using induction. By the convex statement definition of convexity we know this holds true for $k=2$. It obviously holds for $k=1$ as that is equivalent to having each point in C. Now assume it holds for $k-1$.
$\therefore z = \frac{\theta_1 x_1}{\theta_1+\theta_2+\cdots+\theta_{k-1}} + \cdots + \frac{\theta_{k-1}x_{k-1}}{\theta_1+\theta_2+\cdots+\theta_{k-1}} \in C$
and $\therefore y=(\theta_1+\theta_2+\cdots+\theta_{k-1})z+\theta_k x_k \in C$.
which gives us the line from z to x_k. $\therefore$ putting it in the convex set of C hence proven by induction that $\theta_1x_1+\theta_2x_2+\cdots+\theta_kx_k \in C$.
Alternatively: By definition a convex set is a set where, for any two points in the set, the line segment connecting the two is contained within the set. $\therefore$ as C is a convex set with $x_1,\cdots,x_k \in C$, then the line segment between any two points in C is also contained within C hence as $\theta_i \geq 0$ where $\theta_1+\cdots+\theta_k=1$ $\therefore$ $\theta_1x_1+\theta_2x_2+\cdots+\theta_kx_k$ is a convex combination of $x_1,x_2,\cdots,x_k$ hence marking it part of their connecting line segment. As the line segment between any two points in C is also contained within C (by definition) and $\theta_1x_1+\theta_2x_2+\cdots+\theta_kx_k$ is on the line segment of $x_1,x_2,\cdots,x_k$ $\therefore$ it is intuitively in C hence shown.
(4b) To show this, we need to show that $\exists$ a real valued constant $L_t \geq 0$ $\forall x_1, x_2 \in D_1$ we have:
$\star = \|(f_k \circ f_{k-1} \circ \cdots \circ f_1)(x_1)-(f_k \circ f_{k-1} \circ \cdots \circ f_1)(x_2)\| \leq L_t\|x_1-x_2\|$.
where $f_i : D_i \to D_{i+1}$ is L_i-lipschitz continuous $\therefore$
for all $x_1, x_2 \in D_i$, $\exists$ $\|f_i(x_1)-f_i(x_2)\| \leq L_i\|x_1-x_2\|$.
hence by this definition we can manipulate each function in $\star$ to get: $\|f_k(f_{k-1}(\cdots f_1(x_1))) - (f_{k-1}(\cdots f_1(x_2)))\| \leq L_k\|f_{k-1}(f_{k-1}(\cdots f_1(x_1))) - (f_{k-1}(\cdots f_1(x_2)))\|$
$\leq L_k^2\|f_k(f_{k-2}(\cdots f_1(x_1))) - (f_{k-2}(\cdots f_1(x_2)))\|$ and so on $\leq L_k^k\|x_1-x_2\|$ $\therefore$ marking $f_k \circ f_{k-1} \circ \cdots \circ f_1$ an L_t-Lipschitz continuous w/ $L_t = L_k^k$.