MACHINE LEARNING - DATA IN PYTHON

Alon Tsalik Shmilovich - JCE - Software Engineering M.Sc

Research with Python - Final Project

MACHINE LEARNING - DATA IN PYTHON

- In this seminar we'll review several solutions for **classification** problem in Machine Learning using SKlearn:
  - SVM
  - Neural Networks
  - Logistic Regression and SLR
- We will see the differences between the accuracies on those algorithms, k-fold

MACHINE LEARNING - DATA IN PYTHON

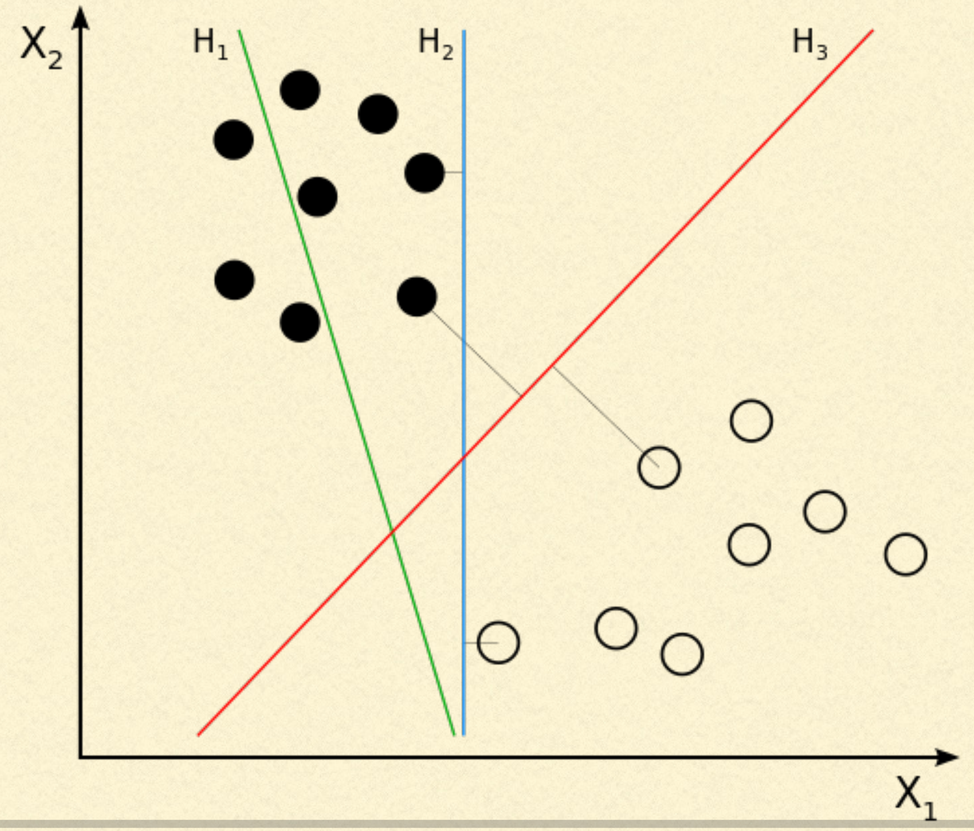
- SKlearn - SciKit Learn
- Used for learning algorithms.
- Python? PYTHON! Because:
  - Strong math libraries.
  - Simplicity in syntax.

SVM - SUPPORT VECTOR MACHINE

- Supervised learning.
- One of ML problems is Classification.
- Given a set of training data (x,y), we would like to build a linear classifier that will determine future data x's y.
- data examples (x,y):
  - x is mail, y is spam or not.
  - x is a medical picture, y says if it's sick or not.

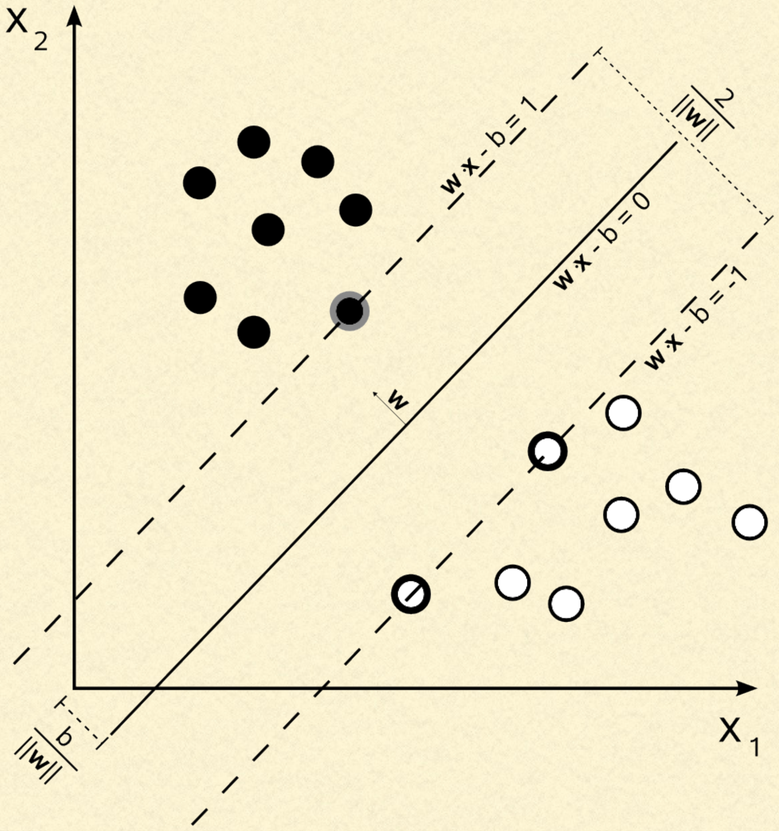
SVM - MATHEMATICAL BACKGROUND

- Our classifier will be linear - we want the best one, with the biggest margin



SVM - MATHEMATICAL BACKGROUND

- Our classifier will be linear - we want the best one, with the biggest margin



SVM - MATHEMATICAL BACKGROUND

- Usually there are 2 classes: A and B, and given sets of m examples of x and their label y, where (x can be a features vector):

$$(x_1,y_1),(x_2,y_2),.....,(x_m,y_m)$$

$$where \begin{cases} y_k=1 & if \ x_k \in class \ A \\ y_k=-1 & if \ x_k \in class \ B \end{cases}$$

- SVM actually constructs a hyperplane in a high-dimentional space that is the classifier.
- A better classifier is the one that takes the biggest margin.

SVM - MATHEMATICAL BACKGROUND

- The high-dimensional space can be defined by (vector) x that satisfies:

$$\vec{x}\ \vec{w} + b = 0$$

- When w is a normal to the space.

- Finally, after mathematical manipulations, the classifier is calculated by minimizing this:

$$\left[\frac{1}{n}\sum max(0, 1 - y_i(\vec{w}\cdot\vec{x_i} - b))\right] + \lambda \|\vec{w}\|^2$$

- Lambda - how wide margin should be

- In programming, we set c to be error penalty.

SVM - MATHEMATICAL BACKGROUND

- Kernel trick is a way taking data from low levels to high
  - From a consumption that in high level, a better classifier will be found.
  - Scattered the original vectors in a sharper way

$$x \longrightarrow \phi(x)$$

$$k(x,y) = \phi(x) \cdot \phi(y)$$

SVM - MATHEMATICAL BACKGROUND

- Kernel example:
  - First we make dot product between x and y.
  - And then power 2 (line 3)
  - Opening brackets will lead to line 4
  - And now we can split it to 2 vectors but in 3D (opposite dot product)

$$k(x,y) = (x \cdot y)^2$$

$$x=\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad y=\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$$

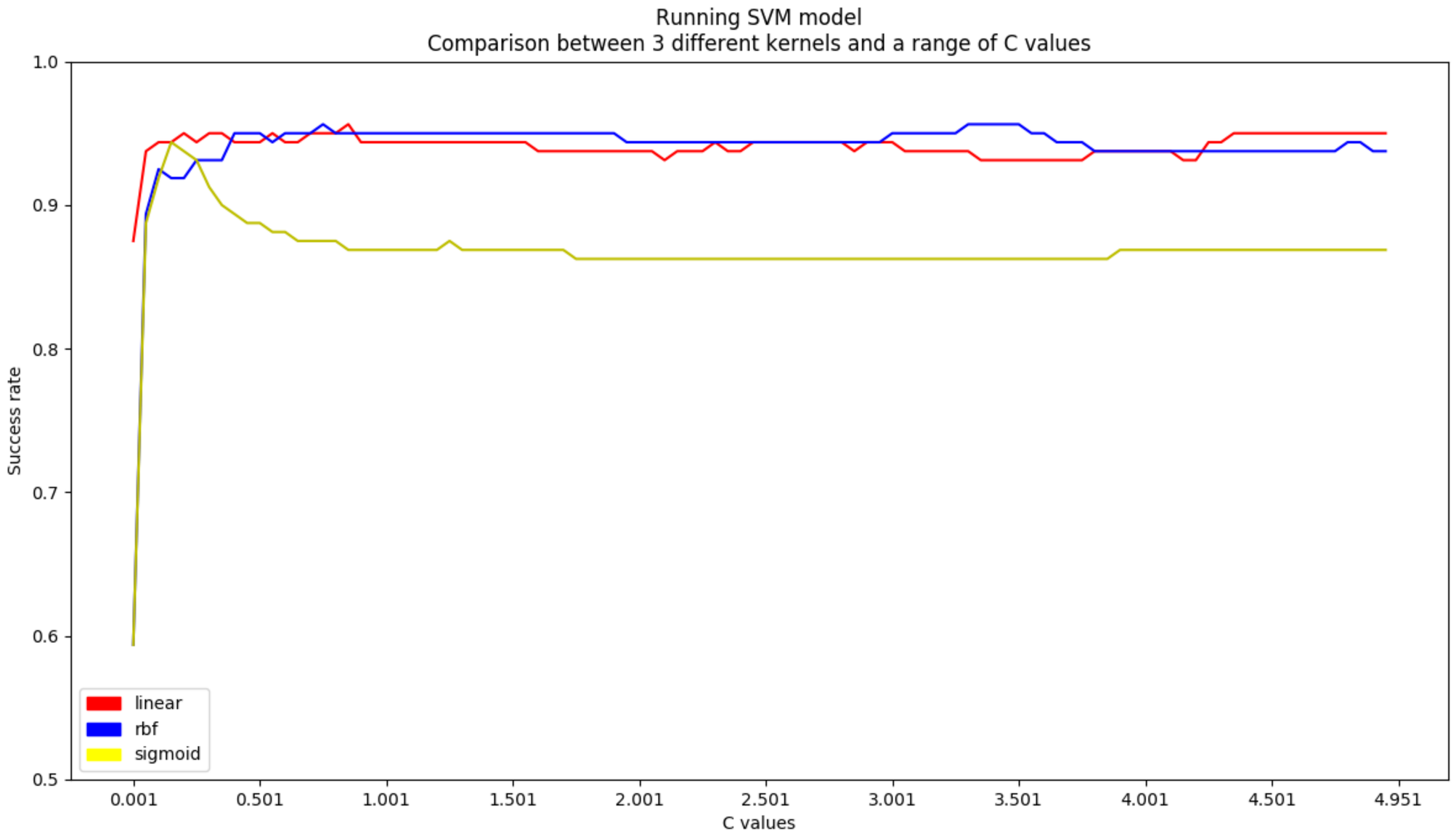
$$k(x,y) = (x_1y_1+x_2y_2)^2$$

$$=x_1^2y_1^2+2x_1y_1x_2y_2+x_2^2y_2^2$$

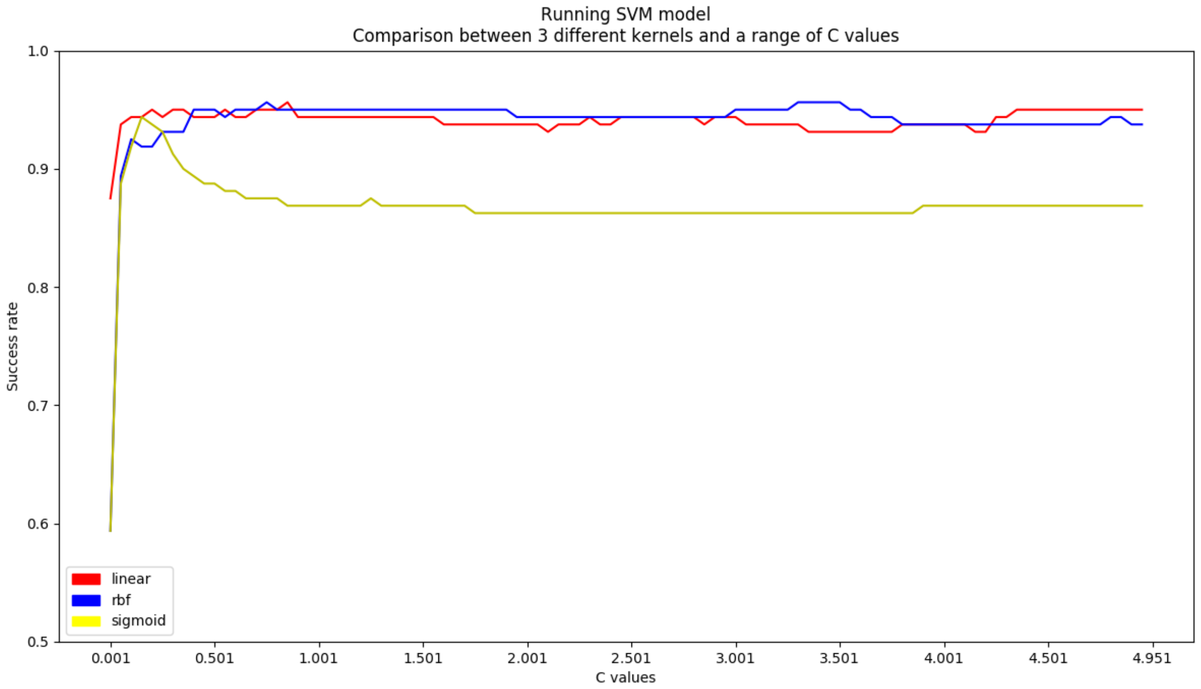
$$=\begin{pmatrix} x_1^2 \\ \sqrt{2}\,x_1x_2 \\ x_2^2 \end{pmatrix} \cdot \begin{pmatrix} y_1^2 \\ \sqrt{2}\,y_1y_2 \\ y_2^2 \end{pmatrix}$$

SVM - IN PYTHON

- Without sklearn library, the implementation is much longer… using numpy and a lot of mathematical calculations…
- In python:
  - By using SVC function, you build a SVM model. clf = SVC()
  - Then using fit function you fit the model to the training data: clf.fit(x_train, y_train)
  - Then we can predict using the predict function: clf.predict(x_test, y_test)
  - And we can see the accuracy of our results by score: clf.score(x_test, y_test)
  - In next slides we'll see graphs that I got from using this algorithms on data that has 21 x features and a y classification.


Running SVM model
Comparison between 3 different kernels and a range of C values
Success rate
C values
1.0
0.9
0.8
0.7
0.6
0.5
0.001
0.501
1.001
1.501
2.001
2.501
3.001
3.501
4.001
4.501
4.951
linear
rbf
sigmoid

SVM - GRAPH



- The plot shows the difference betweeb the accuracies from the predicted results on the test set and the true results.
- 3 kernel functions were used: linear, rbf and sigmoid.
- C values changes - error penalty. Not a lot of changes when getting to c=1
- Sigmoid is less correct, linear and rbf (radial basis function) are better

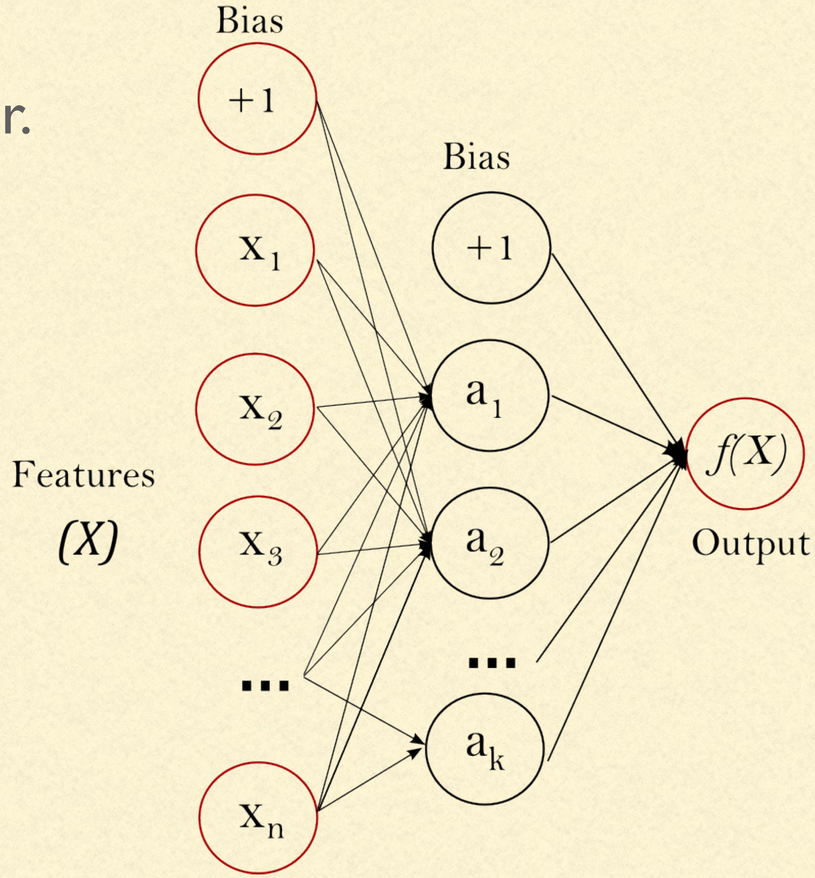
NEURAL NETWORKS MODEL - MLP

- Another way of learning and classifying is using neural networks.
- MLP - Multi Layer Perceptron
- This is the basic way of NN, today there are new kinds of NN such as CNN, dropout…
- Imagine our brain - built from a lot of neurons connected, data is passing through all layers.
- Here, the input set of x with m features (dimensions) and targeted y.
- The first layer is x, connected to the second layers with weights:

$$x_1 w_1 + x_2 w_2 + \ldots + x_m w_m$$

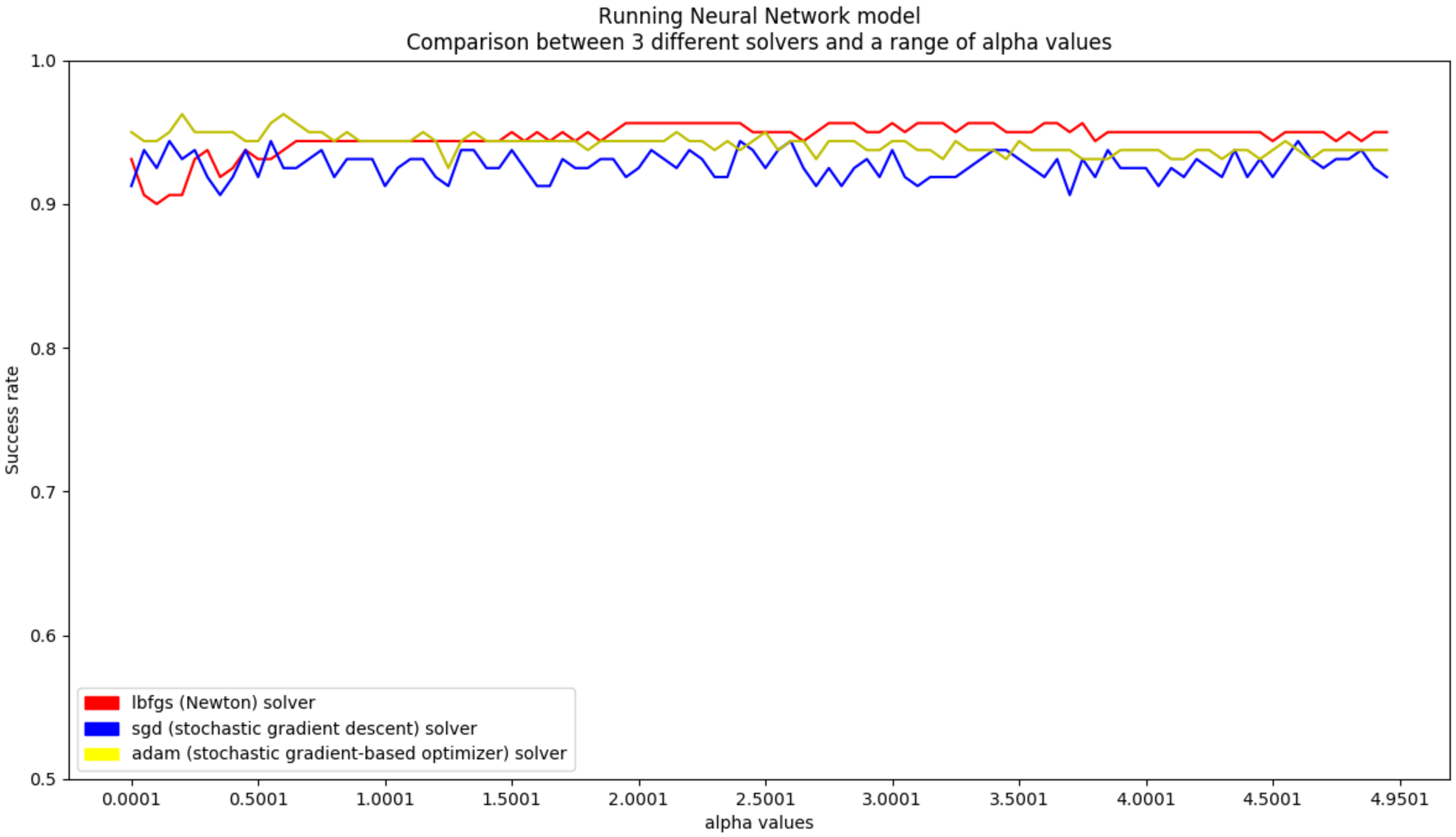
NEURAL NETWORKS MODEL - MLP

- And output is the next layer's input and so on. Those layers are called Hidden Layers.
- This figure has one hidden layer.

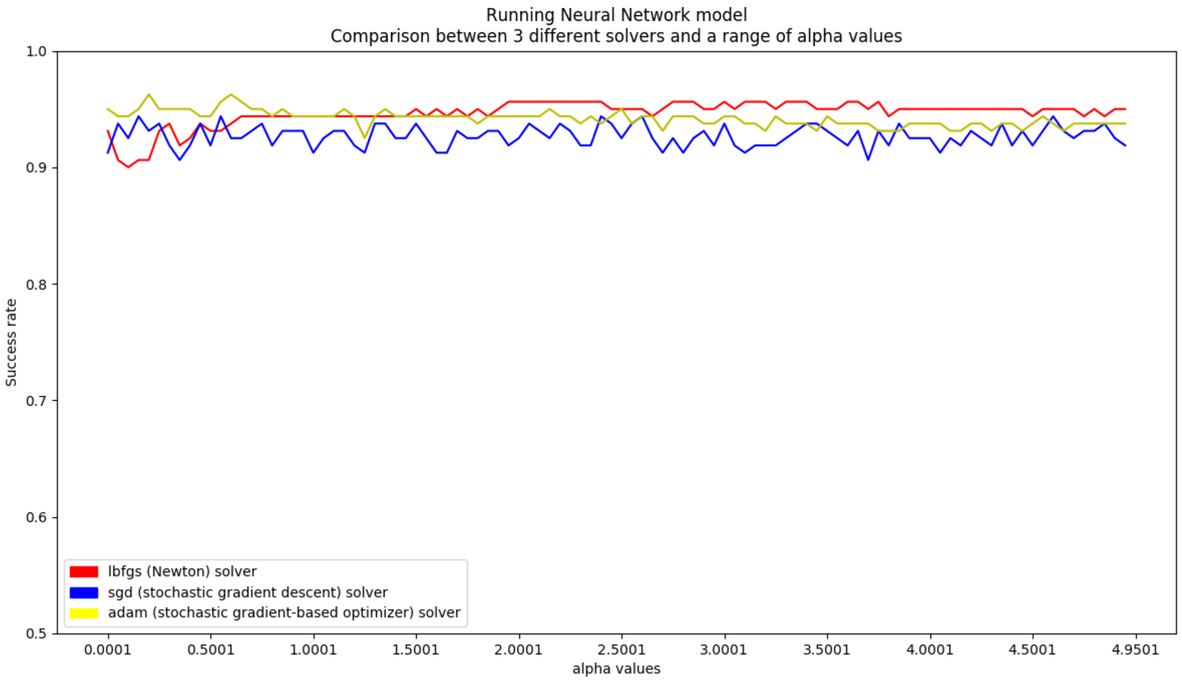


NEURAL NETWORKS - IN PYTHON

- In python:
  - Here also, implementation with sklearn is pretty easy: clf = MLPClassifier()
  - Using clf.fit(x_train, y_train) fits our model
  - And clf.score(x_test, y_test) measures our accuracy in predicting the test sets.


Running Neural Network model
Comparison between 3 different solvers and a range of alpha values
1.0
0.9
0.8
0.7
0.6
0.5
Success rate
lbfgs (Newton) solver
sgd (stochastic gradient descent) solver
adam (stochastic gradient-based optimizer) solver
0.0001
0.5001
1.0001
1.5001
2.0001
2.5001
3.0001
3.5001
4.0001
4.5001
4.9501
alpha values

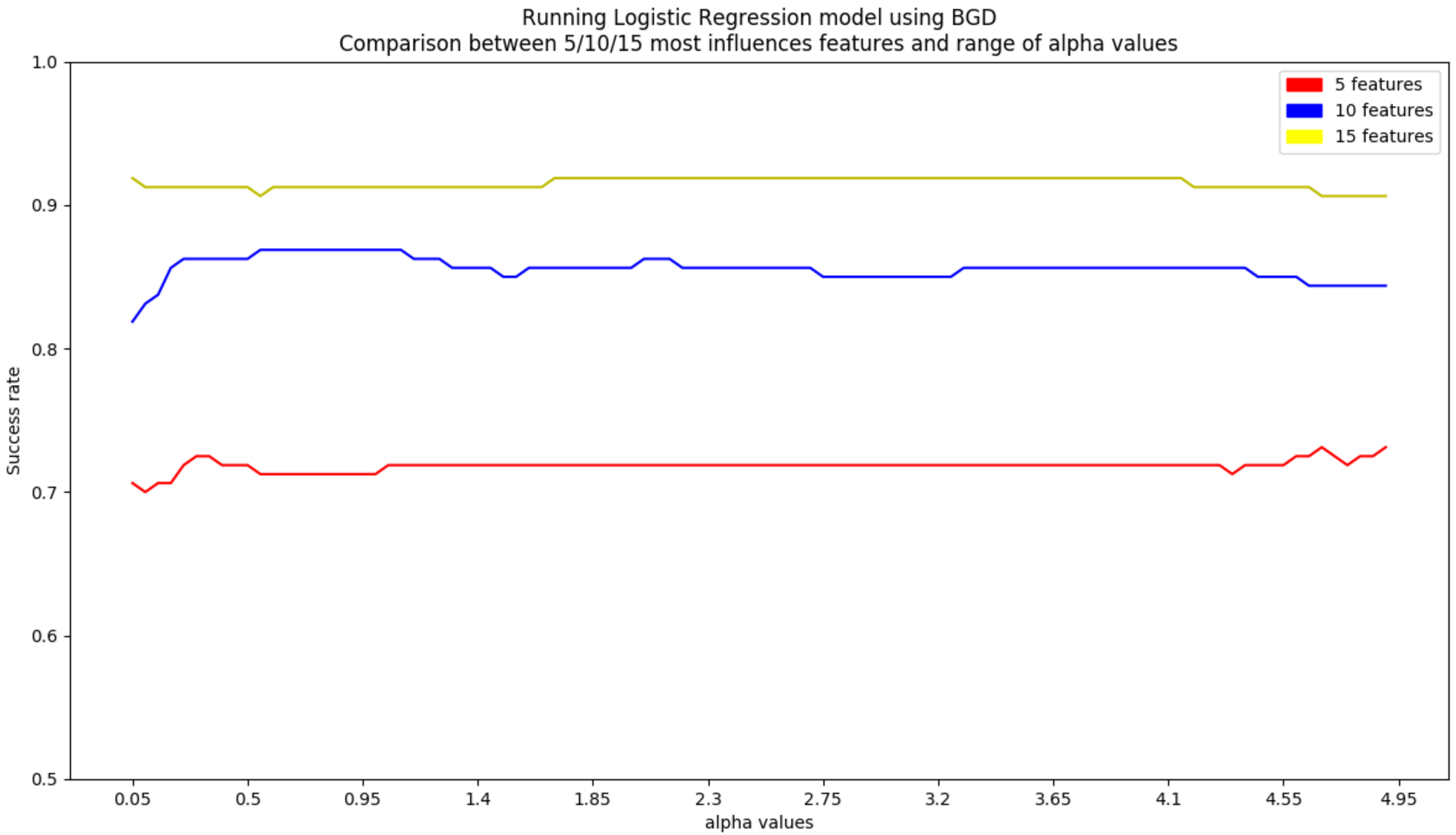
NEURAL NETWORK - GRAPH

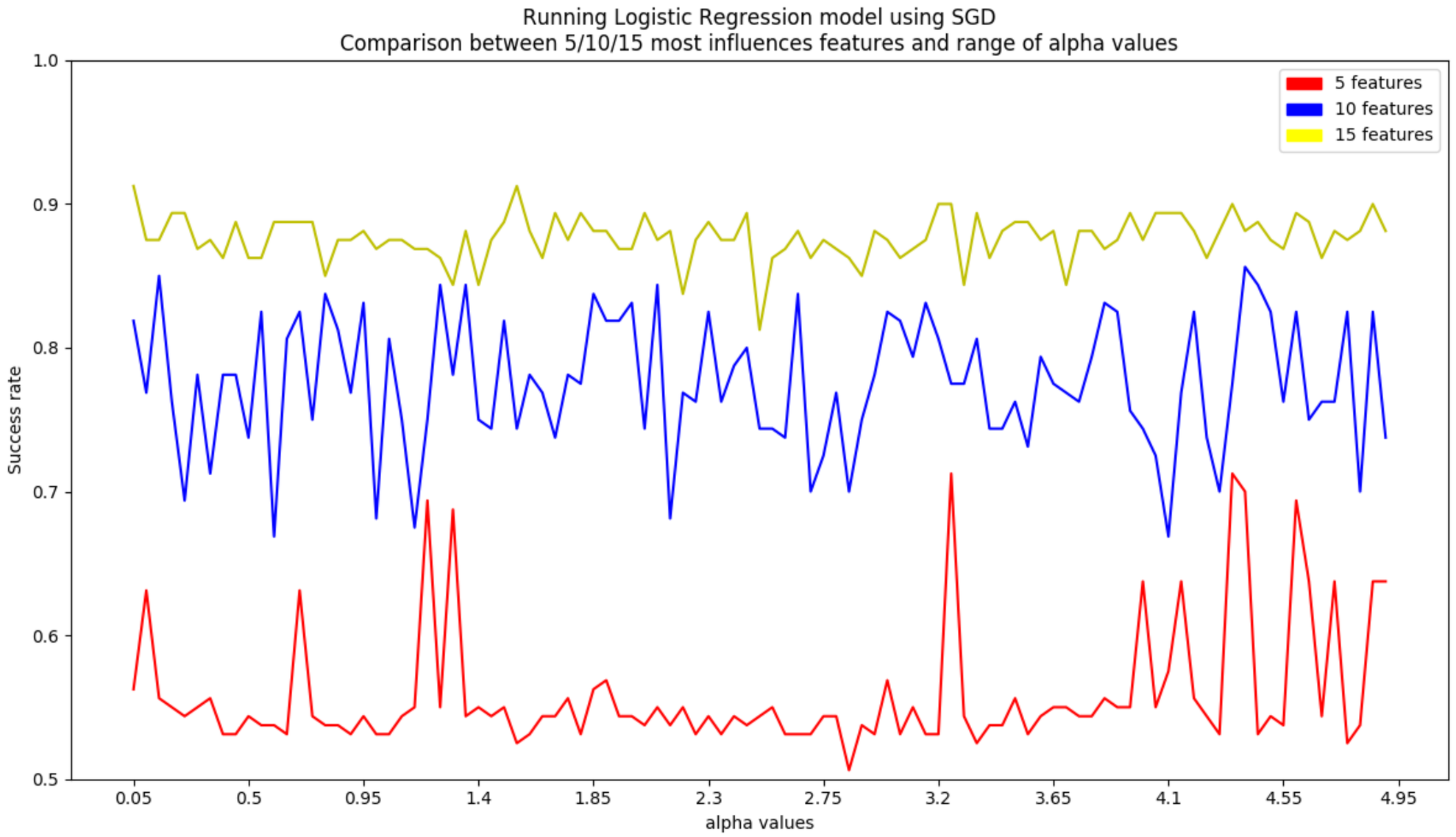


- The plot shows the difference between the accuracies from the predicted results on the test set and the true results.
- 3 solvers were used: lbfgs, sgd, adam.
- **alpha is a regularization param for fitting the model.**
- Depends on data's length: for small - better use lbfgs, bigger (k's) - adam.

LOGISTIC REGRESSION

- Logistic Regression is a statistical method to analyse datasets.
- Input data may have one or more variable (features), and an outcome.
- Helps to predict
- Stochastic Logistic Regression is the same method, but with simplified calculations.


Running Logistic Regression model using BGD
Comparison between 5/10/15 most influences features and range of alpha values
5 features
10 features
15 features
Success rate
1.0
0.9
0.8
0.7
0.6
0.5
0.05
0.5
0.95
1.4
1.85
2.3
2.75
3.2
3.65
4.1
4.55
4.95
alpha values


Running Logistic Regression model using SGD
Comparison between 5/10/15 most influences features and range of alpha values
5 features
10 features
15 features
Success rate
alpha values
1.0
0.9
0.8
0.7
0.6
0.5
0.05
0.5
0.95
1.4
1.85
2.3
2.75
3.2
3.65
4.1
4.55
4.95

LOGISTIC REGRESSION GRAPHS

- We saw that the more features x has, the more y prediction is accurate.
- Using SGD makes “noise”:
  - The accuracy is not stable
  - Using less features gives us very bad accuracy.
- Alpha is a regularization parameter that helps us fit the data to the model, “shrink” it.

K-FOLD CROSS VALIDATION

- It is used to split data for train and test sets. The goal is to overcome **overfitting**
- Usually - 80% train, 20% test, but it can be also 60-40, 70-30 etc.
  - This is data cross validation.
- We can also divide the data into k sets, and use (k-1) for train.
- Then running k times the algorithm we want.
- This is called k-fold cross validation.

SKLEARN ADVATANGES

- SKlearn advantages:
  - Works nicely with other inner python libraries such as numpy, scipy.
  - Good documentation, understandable.
  - Easy to use.

SKLEARN DISADVANTAGES

- SKlearn disadvantages:
  - Data can be changes very quickly. Doesn't support updating data bases - learning is on data and no more.
  - SKlearn does not support GPU support - VERY BIG CALCULATIONS, such as for images etc.
  - Data sensitivty - "learnable data"

SOURCES

- SKlearn documentation: http://scikit-learn.org/
- Mathematical graphs SVM - Wikipedia.
- Boser, Guyon, Vapnik: A Training Algorithm for Optimal margin Classifiers, 1992, Proceedings of the fifth annual Workshop Conceptual learning theory, pages 144-152.