

# ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ ΣΤΗΝ ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

## **Στατιστική Μάθηση-Υπολογιστική Νοημοσύνη**

**Σουράνης Παναγιώτης**

**AEM:17**

.....

Το παρακάτω κείμενο είναι μια σύντομη αναφορά της εφαρμογής του αλγορίθμου SVM στο Dataset Breast cancer το οποίο πάρθηκε από την ιστοσελίδα:

[https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic))

### **Περιγραφή Dataset:**

Στο παρακάτω σύνολο δεδομένων μας δίνονται τα χαρακτηριστικά τα οποία συλλέχθηκαν μέσω μίας παρακέντησης με λεπτή βελόνη στην μάζα του μαστού.

Τα χαρακτηριστικά τα οποία αναφέρονται είναι για παράδειγμα ( Η περίμετρος της μάζας, η ομαλότητα (smoothness), η υφή , η συμμετρία και λοιπά).

Η διάγνωση η οποία προέκυψε αναφέρεται σε καλοήγη (Benign) και κακοήγη (Malignant) μάζες του μαστού.

Στόχος μας λοιπόν είναι ο αλγόριθμος να προβλέπει όσο καλύτερα γίνεται αν η μάζα του μαστού είναι καλοήγη ή κακοήγη.

## Ανάλυση Dataset:

Ας αρχίσουμε λοιπόν πρώτα με την ανάλυση των χαρακτηριστικών του συνόλου δεδομένων που μας δόθηκε

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280



concavity_mean	concave points_mean	...	texture_worst	perimeter_worst	area_worst	smoothness_worst	compactness_worst	concavity_worst	concave points_worst
0.3001	0.14710	...	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654
0.0869	0.07017	...	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860
0.1974	0.12790	...	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430
0.2414	0.10520	...	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575
0.1980	0.10430	...	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625

symmetry_worst	fractal_dimension_worst	Unnamed: 32
0.4601	0.11890	NaN
0.2750	0.08902	NaN
0.3613	0.08758	NaN
0.6638	0.17300	NaN
0.2364	0.07678	NaN

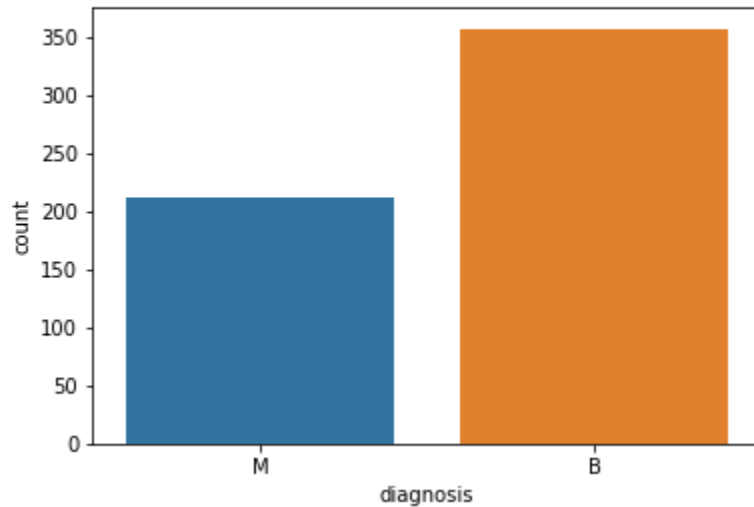


Αυτή είναι μια γενική εικόνα των χαρακτηριστικών. Μπορούμε να παρατηρήσουμε εξ' αρχής ότι έχουμε μια κατηγορία χαρακτηριστικών η οποία είναι κενή και δεν μας δίνει κάποια ιδιαίτερη πληροφορία, αυτή είναι η "Unnamed 32" οπότε πρέπει να αφαιρεθεί από το dataset.

Επίσης η στήλη η κλάση Id δεν μπορεί να μας παρέχει κάποια χρήσιμη πληροφορία η οποία θα συνεισφέρει στην διάγνωση οπότε και αυτή θα πρέπει να αφαιρεθεί.

Ας δούμε λοιπόν στην συνέχεια την κλάση της διάγνωσης “diagnosis”

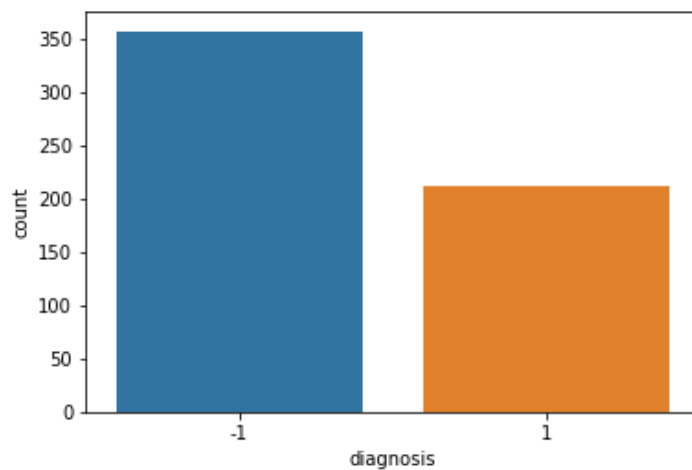
Number of Benign: 357  
Number of Malignant : 212



Σχήμα 1.1

Παρατηρούμε λοιπόν ότι στα δεδομένα μας το **37.25%** των μαζών ήταν κακοήθεις και το **62.75%** καλοήθεις.

Επειδή τα δεδομένα μας δεν είναι αριθμητικά αλλά είναι κατηγορικά θα πρέπει να τα μετατρέψουμε σε binary προκειμένου στην συνέχεια να εφαρμόσουμε τον αλγόριθμο SVM.



Σχήμα 1.2

Η κλάση -1 λοιπόν αναφέρεται τώρα στην κλάση Benign και η κλάση +1 στην κλάση Malignant.

Ας δημιουργήσουμε λοιπόν στην συνέχεια έναν πίνακα Συσχέτισης (Correlation Matrix) για να παρατηρήσουμε την συσχέτιση μεταξύ των χαρακτηριστικών.

Σχήμα 1.3



Όπως βλέπουμε υπάρχουν αρκετά χαρακτηριστικά όπως για παράδειγμα (perimeter worst,area mean) ή (perimeter worst,area worst) και λοιπά τα οποία έχουν μεγάλη συσχέτιση μεταξύ τους οπότε μας παρέχουν την ίδια πληροφορία και για αυτό πρέπει να επιλέξουμε ποια θα κρατήσουμε και ποια θα απαλειφθούν.

Πρωτού όμως προχωρήσουμε παρακάτω θα πρέπει να υλοποιήσουμε ένα normalization η αλλιώς standardization διότι έχουμε χαρακτηριστικά για τα οποία οι τιμές τους διαφέρουν πολύ.

Για την κανονικοποίηση χρησιμοποιήθηκε η κανονική κατανομή  $N(0,1)$ .

$(X-\mu)/\sigma$  όπου  $\mu$  η μέση τιμή και  $\sigma$  η διακύμανση standart deviation

Θα μπορούσε επίσης να χρησιμοποιηθεί και η μέθοδος min – max

Στην συνέχεια παρουσιάζουμε μερικά θηκογράμματα τα οποία είναι ένας βολικός τρόπος γραφικής αναπαράστασης μιας μεταβλητής,ως προς πέντε βασικές παραμέτρους οι οποίες συνοψίζουν την κατανομή της.

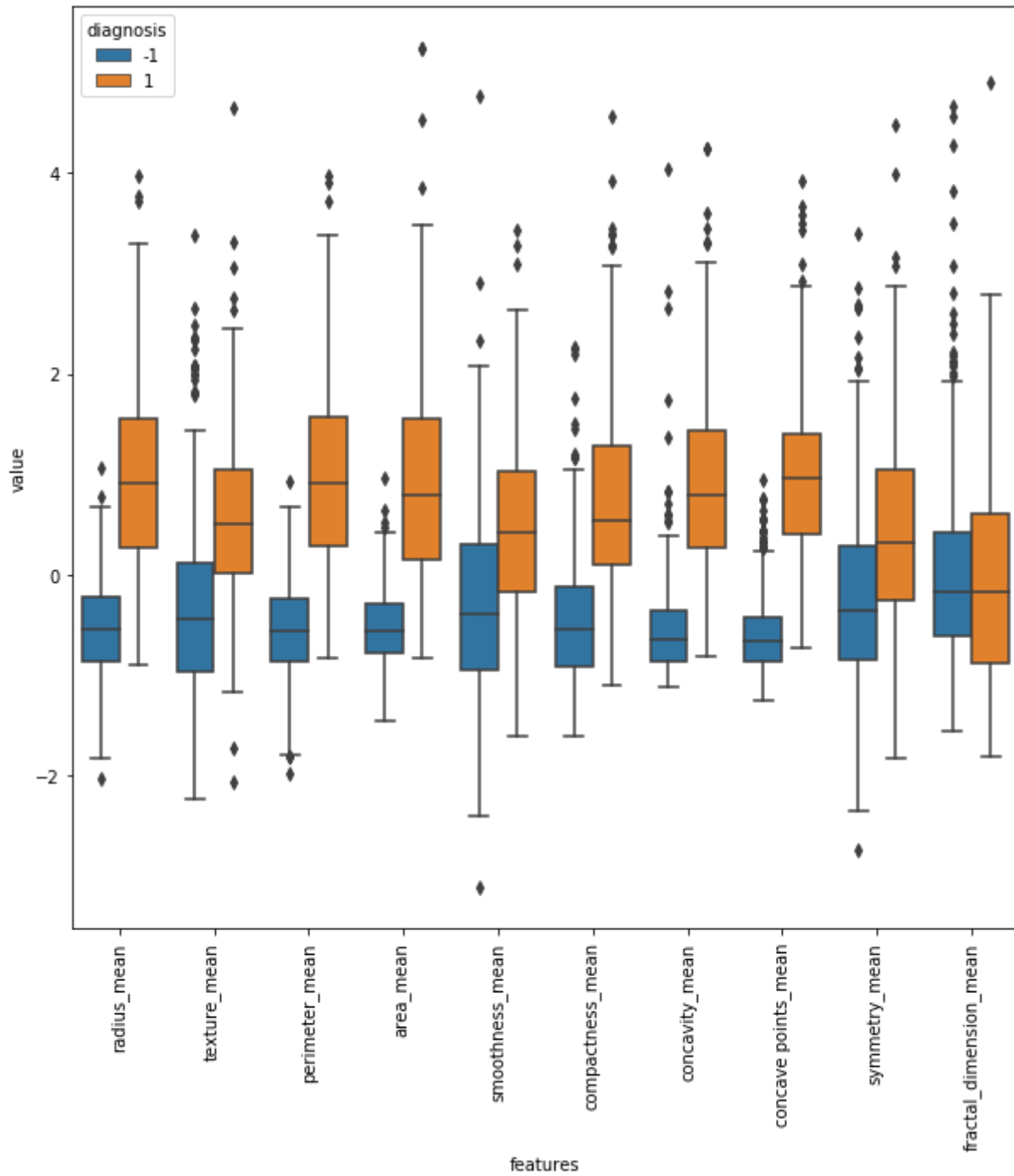
Αυτές είναι

- Ελάχιστη τιμή (min)
- 1<sup>ο</sup> τεταρτημόριο (Q1)
- Διάμεσος (Q2)
- 3<sup>ο</sup> τεταρτημόριο (Q3)
- Μέγιστη τιμή (max)

Επίσης τα θηκογράμματα μας αναπαριστούν που βρίσκεται το 50% των τιμών και ακόμη μας παρουσιάζουν τις ακραίες τιμές που βρέθηκαν.

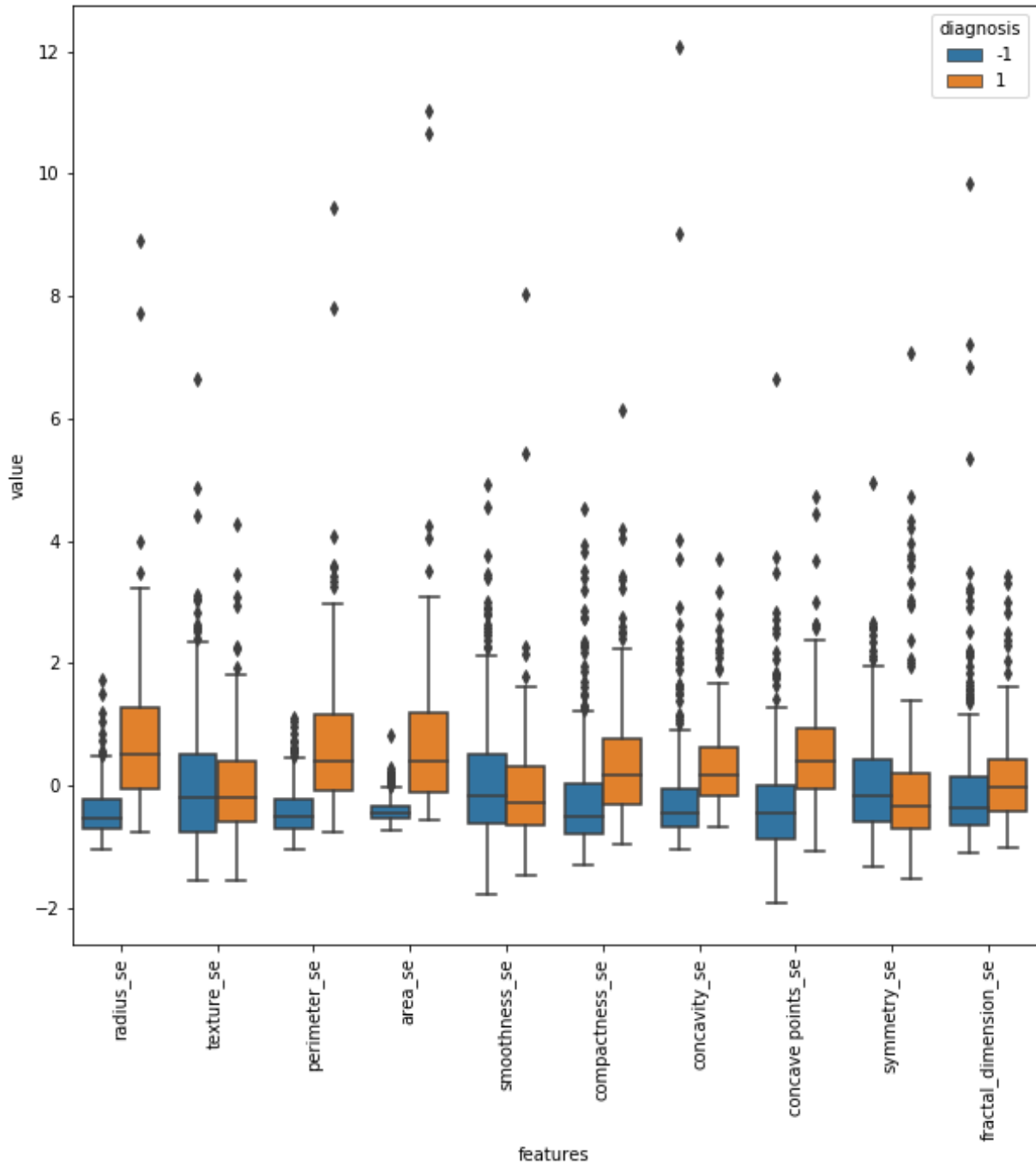
Οι μεταβλητές που αναφέρονται είναι οι 10 πρώτες.

Σχήμα 1.4



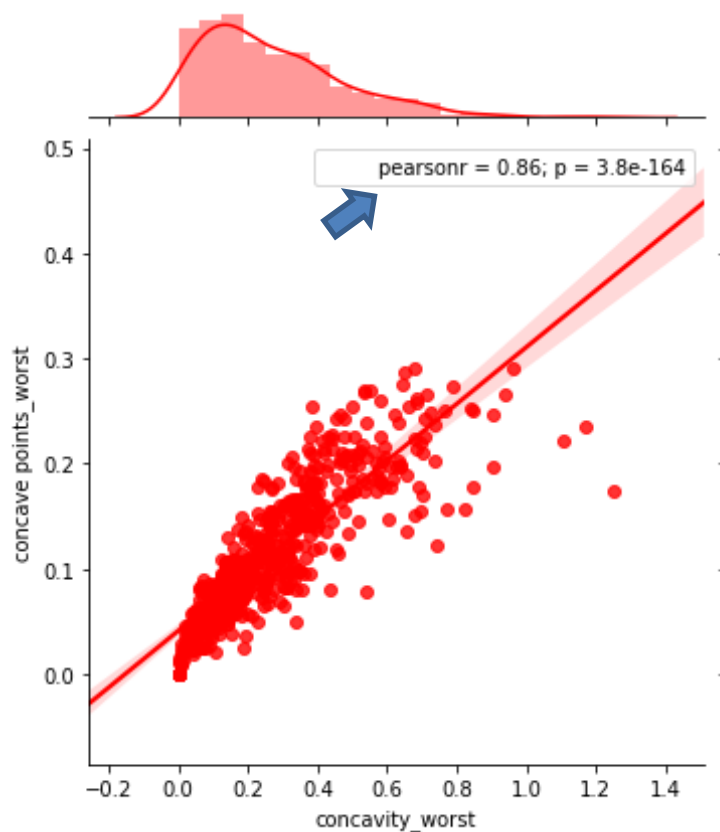
Οι μεταβλητές που αναφέρονται είναι από 10-20.

Σχήμα 1.5



Αυτό που θα μπορούσαμε επίσης να κάνουμε είναι να σχεδιάσουμε ένα κοινό διάγραμμα το οποίο θα μας δίνει μια οπτικοποίηση για τα αποτελέσματα που μας έδωσε ο πίνακας συσχέτισης για τα την συσχέτιση μεταξύ 2 χαρακτηριστικών.

Ας πάρουμε για παράδειγμα τις μεταβλητές (concavity worst και concave points worst) οι οποίες όπως είδαμε από τον πίνακα έχουν συντελεστή συσχέτισης 0.9



Σχήμα 1.6

Όπως βλέπουμε ο συντελεστής συσχέτισης Pearson είναι κοντά στην μονάδα που οδηγεί στο συμπέρασμα ότι αυτές οι δύο μεταβλητές έχουν απόλυτη γραμμική συσχέτιση.

Αφού είδαμε λοιπόν ότι πολλές μεταβλητές συσχετίζονται μεταξύ τους τώρα αυτό που μένει είναι να κρατήσουμε αυτές που είναι σημαντικότερες και κατέχουν δηλαδή την περισσότερη πληροφορία σχετικά με την διάγνωση.



Υπάρχουν αρκετοί τρόποι για να γίνει αυτό αλλά αυτοί που θα σχολιαστούν είναι οι RFE ( Recursive Feature Selection) και PCA (Principal Component Analysis)

Τι είναι όμως οι **RFE** και **PCA** ?

## **PCA**

Η μέθοδος PCA (Ανάλυση Κύριων Συνιστωσών), αποτελεί μία γραμμική μέθοδο συμπίεσης Δεδομένων η οποία συνίσταται από τον επαναπροσδιορισμό των συντεταγμένων ενός συνόλου δεδομένων σε ένα άλλο σύστημα συντεταγμένων το οποίο θα είναι καταλληλότερο στην επικείμενη ανάλυση δεδομένων. Αυτές οι νέες συντεταγμένες είναι το αποτέλεσμα ενός γραμμικού συνδυασμού προερχόμενου από τις αρχικές μεταβλητές και εκπροσωπούνται σε ορθογώνιο άξονα, ενώ τα επικείμενα σημεία διατηρούν μια φθίνουσα σειρά όσο αφορά στη τιμή της διακύμανσής τους. Για το λόγο αυτό, το πρώτο κύριο συστατικό (principal component) διατηρεί περισσότερες πληροφορίες δεδομένων σε σύγκριση με το δεύτερο το οποίο δεν διατηρεί πληροφορίες οι οποίες έχουν εισέλθει νωρίτερα (στο πρώτο συστατικό). Τα principal components δεν συσχετίζονται. Η συνολική ποσότητα των principal components είναι ίση με τη ποσότητα των αρχικών μεταβλητών και παρουσιάζει τις ίδιες πληροφορίες στατιστικής. Εντούτοις, η συγκεκριμένη μέθοδος επιτρέπει την μείωση του συνόλου των μεταβλητών, καθώς τα πρώτα συστατικά (principal components) διατηρούν περισσότερο από το 90% των στατιστικών δεδομένων από τα αρχικά δεδομένα.

- Πηγή  
[https://eclass.uoa.gr/modules/document/file.php/DI367/%CE%A5%CE%BB%CE%B9%CE%BA%CF%8C/PCA\\_method.pdf](https://eclass.uoa.gr/modules/document/file.php/DI367/%CE%A5%CE%BB%CE%B9%CE%BA%CF%8C/PCA_method.pdf)

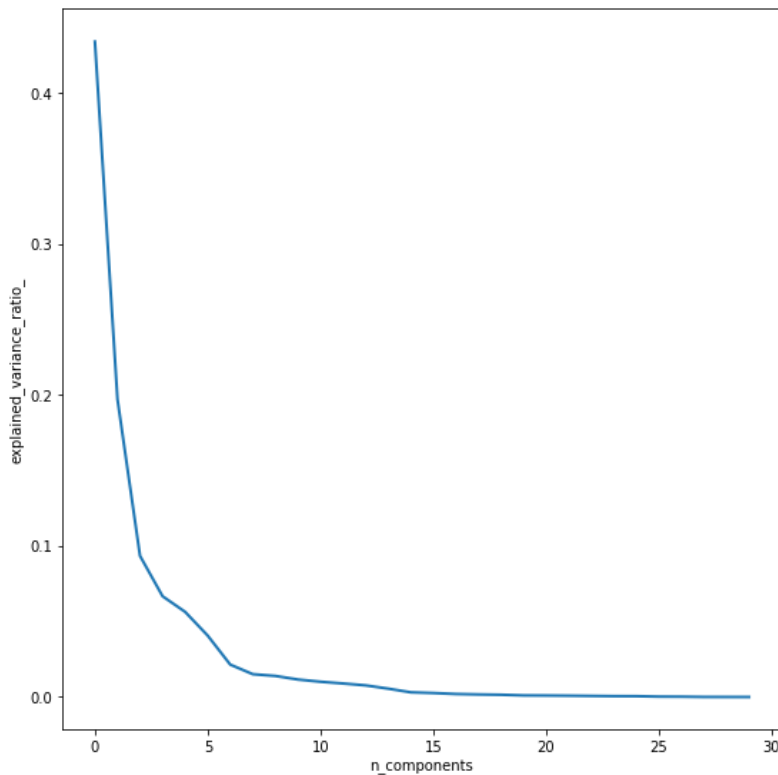
## **RFE (Recursive Feature Elimination)**

Η μέθοδος RFE όπως μας αναφέρει και ο τίτλος της αυτό που κάνει είναι να εξαλείφει χαρακτηριστικά βάσει ενός μοντέλου και με τα υπολοιπόμενα χαρακτηριστικά να υπολογίζει την ακρίβεια του μοντέλου.

Η μέθοδος RFE έχει την ικανότητα να εντοπίζει τα χαρακτηριστικά (attributes) με την περισσότερη συνεισφορά στην πρόβλεψη της κλάσης στόχου.

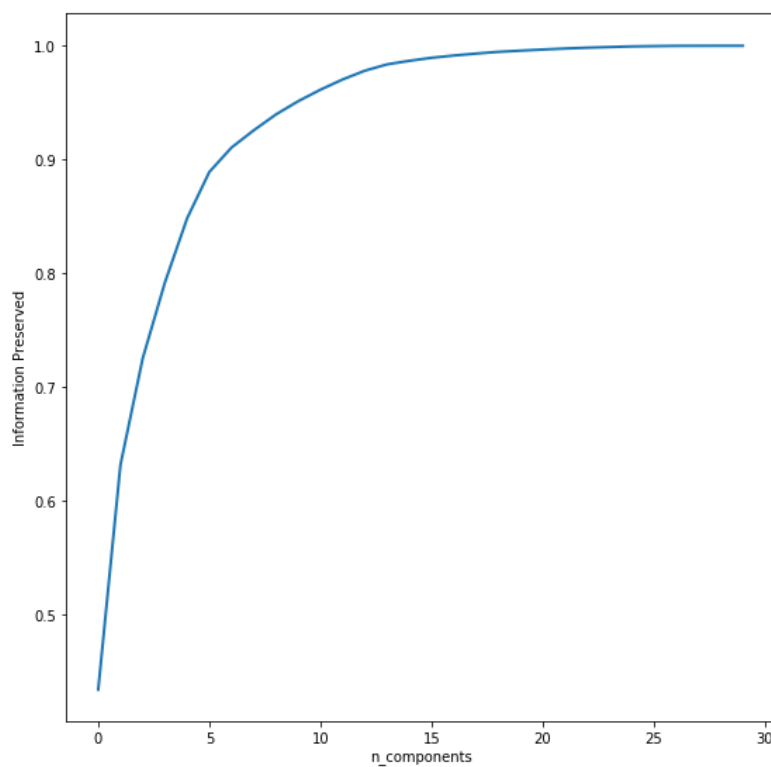
- Πηγή  
<https://medium.com/@aneesha/recursive-feature-elimination-with-scikit-learn-3a2cbdf23fb7>

Ας δούμε λοιπόν μια προσομοίωση κατά πόσο με την μέθοδος PCA μειώνεται η πληροφορία που έχει component όσο αυξάνουμε το πλήθος τους.



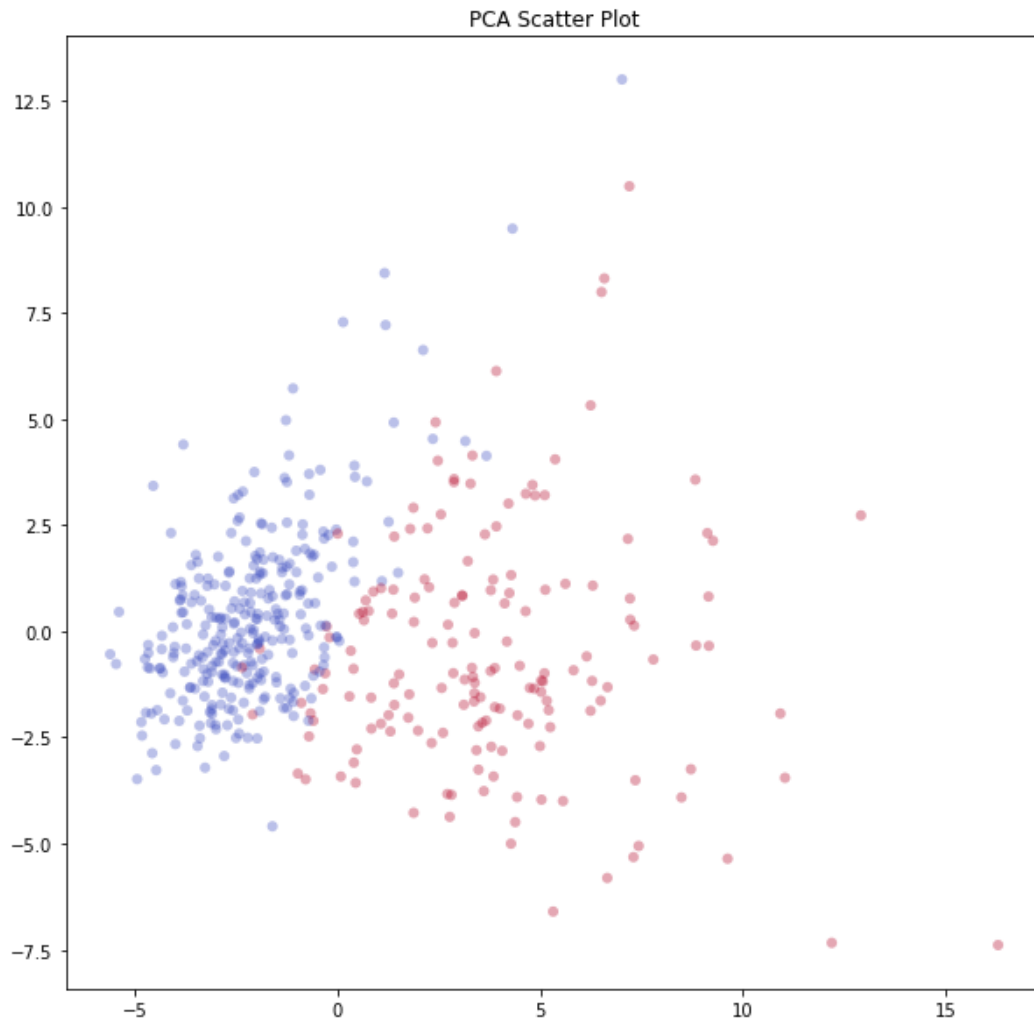
**Σχήμα 1.7**

Παρ'όλα αυτά το μέγεθος της πληροφορίας που διατηρείται όπως φαίνεται και στο κάτω σχήμα αυξάνεται όσο αυξάνεται το πλήθος των components που κρατάμε



**Σχήμα 1.8**

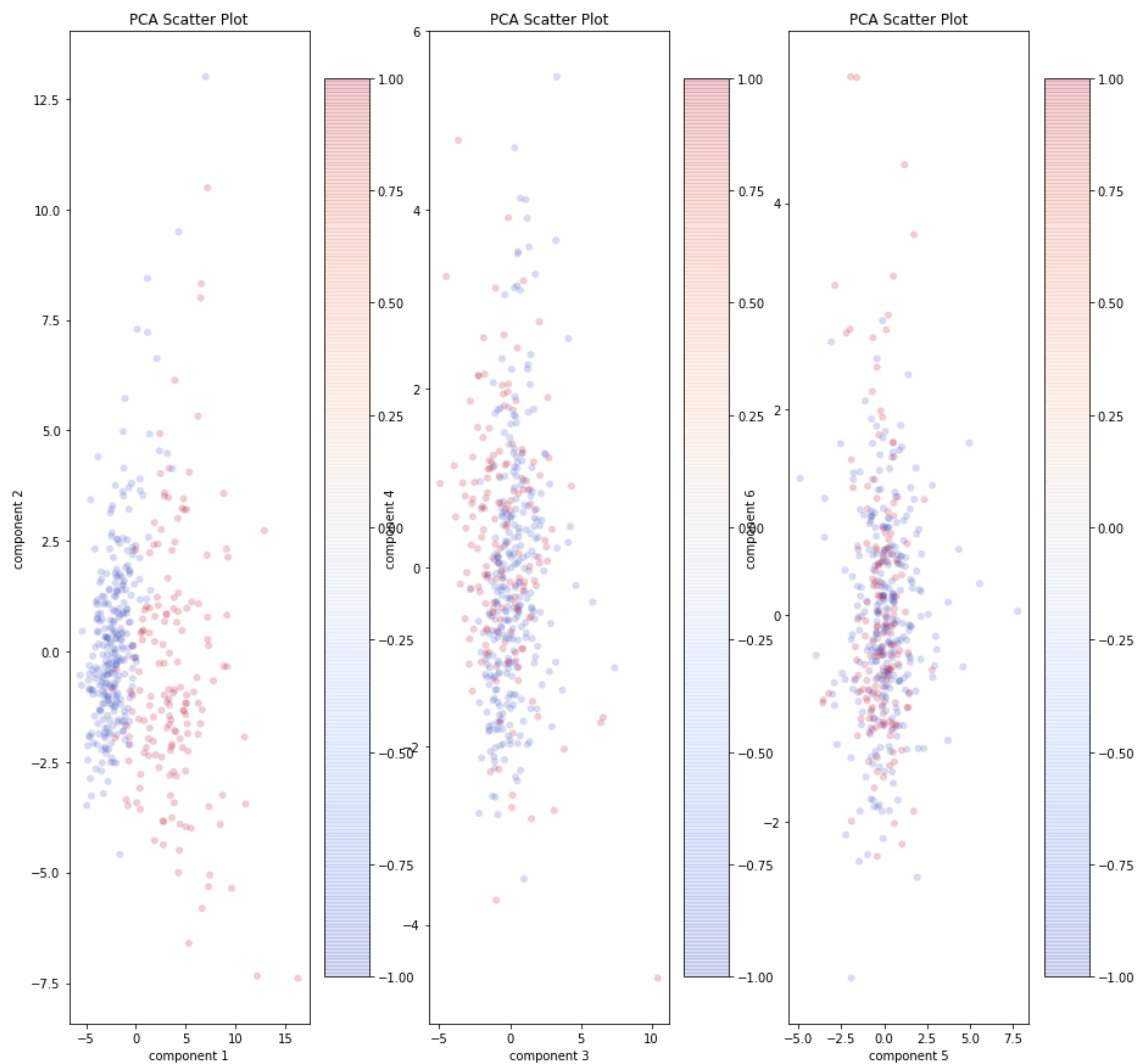
Ας δούμε μια εικόνα του PCA για τα πρώτα 2 χαρακτηριστικά.



**Σχήμα 1.9**

Οι μπλέ κουκίδες αναφέρονται στο πρώτο χαρακτηριστικό το οποίο κράτησε το PCA και οι κόκκινες στο δεύτερο.

Ας δούμε όμως συγκεντρωτικά και για μερικά από τα υπόλοιπα χαρακτηριστικά.



**Σχήμα 1.10**

Αναλυτικά οι διακυμάνσεις για το κάθε χαρακτηριστικό είναι:

PCA Explained Variance Ratio	
Component 1	0.434308
Component 2	0.197401
Component 3	0.093518
Component 4	0.066777
Component 5	0.056425
Component 6	0.040716
Component 7	0.021493

Το άθροισμα τους μας δίνει : **0.9106363829667495** . Άρα έχουμε στη διάθεση μας το **91%** της αρχικής πληροφορίας.

Αφου πραγματοποιήσαμε λοιπόν και την διαδικασία PCA ήρθε η ώρα να προχωρήσουμε στην εκπαίδευση του μοντέλου μας με τον αλγόριθμο SVM.

Οι πυρήνες Kernels που χρησιμοποιήθηκαν είναι ( **Linear Kernel** , **Gaussian Kernel (RBF)** , **Polynomial Kernel** )

Ακόμη έγινε αναζήτηση σε πλέγμα ( Grid Search ) προκειμένου να βρεθεί η καλύτερη παράμετρος C.

### ΣΗΜΕΙΩΣΗ!

Να αναφέρουμε επίσης ότι έγινε αναζήτηση και για τις μεταβλητές sigma για την περίπτωση του Gaussian Kernel και degree of polynomial στην περίπτωση του πολυωνυμικού αλλά κρατήθηκαν μόνο οι βέλτιστες για να κάνουμε αναζήτηση στην συνέχεια για την τιμή του C.

### Linear Kernel (After PCA)

Οι τιμές C στις οποίες έγινε αναζήτηση ήταν:

	0	1	2	3	4	5	6	7	8	9	10
Values for C	0.125	0.25	0.5	1.0	2.0	3.0	4.0	5.0	6.0	8.0	16.0

Οι χρόνοι που χρειάστηκαν για να εκτελεστεί για καθένα από τα C ο αλγόριθμος SVM ήταν:

	0	1	2	3	4	5	6	7	8	9	10
Time Needed to finish	0.349561	0.332236	0.397635	0.35304	0.403338	0.464525	0.463817	0.443742	0.561379	0.459011	0.545618



Βλέπουμε ότι ο γρηγορότερος αλγόριθμος για να ολοκληρωθεί ήταν ο δεύτερος που αντιστοιχούσε στην τιμή του **C=0.25**

### **Precision , Recall , F1**

Πρίν προχωρήσουμε ας αναφέρουμε πως μετρούνται τα Precision, Recall, F1 Score

Έχουμε λοιπόν

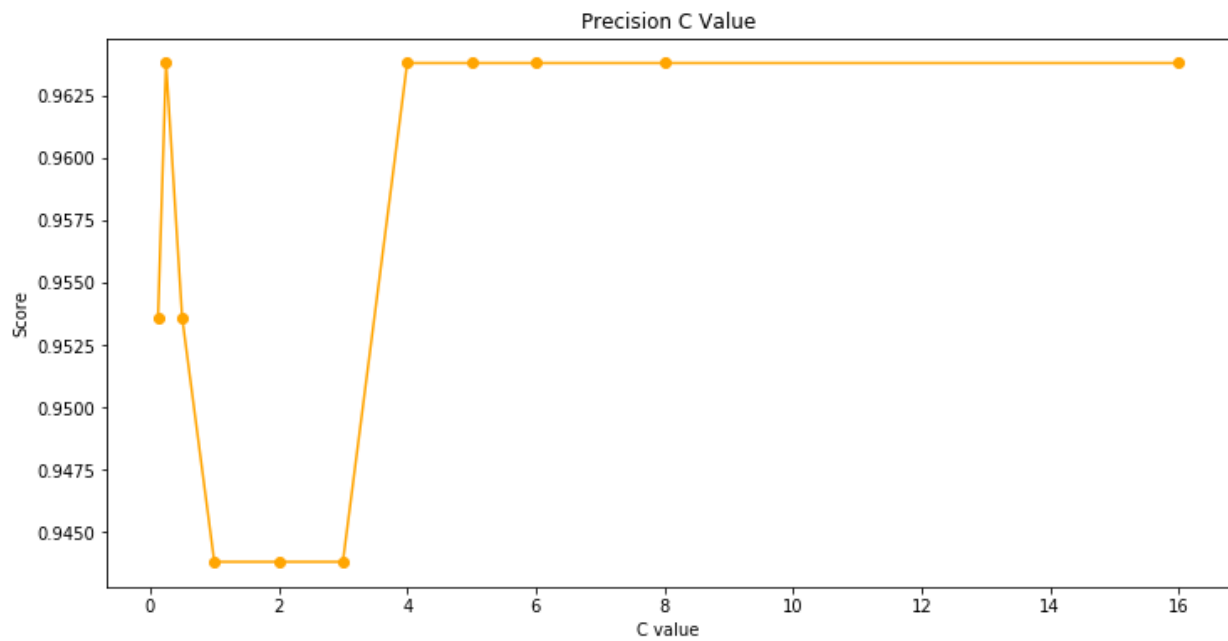
- True positive = correctly identified
- False positive = incorrectly identified
- True negative = correctly rejected
- False negative = incorrectly rejected

**Precision**= $TP/(TP+FP)$

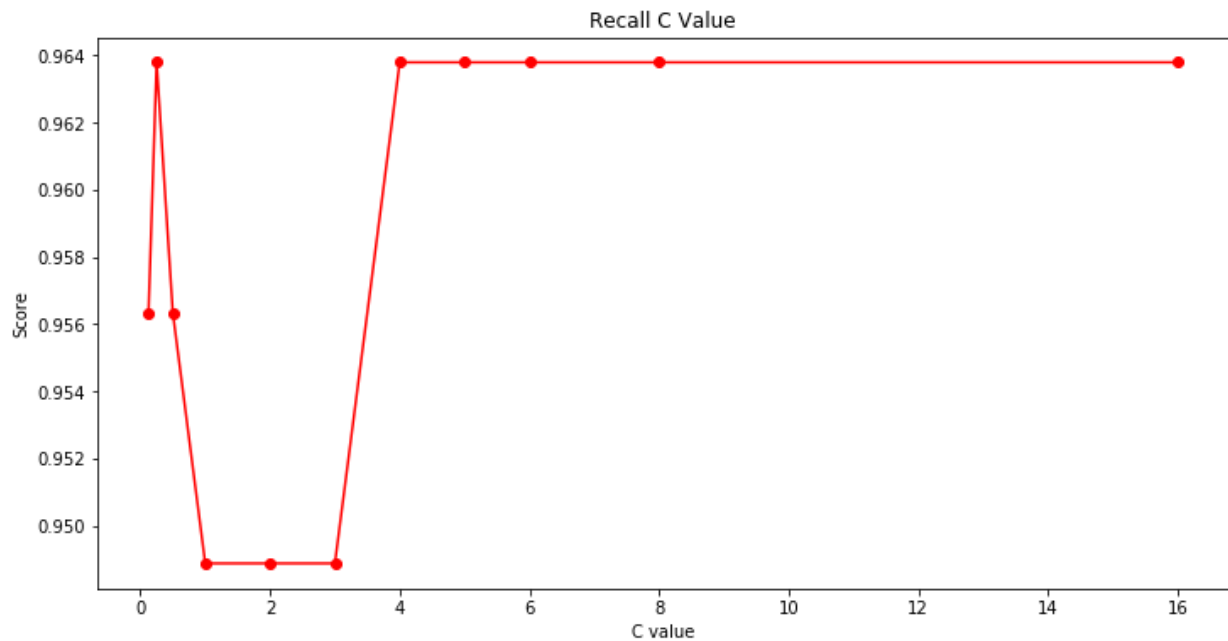
**Recall**= $TP/(TP+FN)$

**F1 Score**= $2*Precision*Recall/(Precision+Recall)$  η αλλιώς θα μπορούσαμε να πούμε ότι είναι ο αρμονικός μέσος μεταξύ του Precision και του Recall.

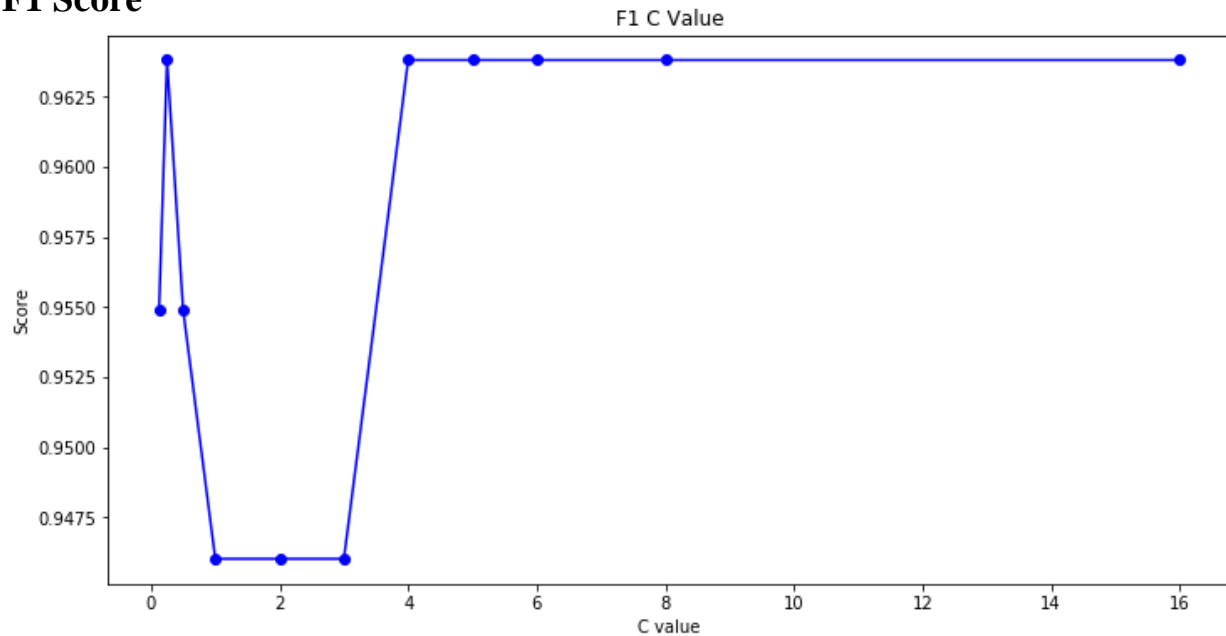
## Precision



## Recall



## F1 Score



Η καλύτερη τιμή βρέθηκε για την τιμή του  $C=0.25$ . Ας δούμε λοιπόν ένα αναλυτικό classification report για την παραπάνω τιμή

	precision	recall	f1-score	support
-1.0	0.97	0.97	0.97	67
1.0	0.96	0.96	0.96	47
avg / total	0.96	0.96	0.96	114

Επίσης τα support vectors που χρειάστηκαν ήταν

**45 support vectors out of 455 points**

## Gaussian Kernel ( RBF )

Θα χρησιμοποιήσουμε για αρχή την τιμή για το **sigma = 3** (variance)

Οι τιμές για την αναζήτηση σε πλέγμα παραμένουν οι ίδιες.

Ας δούμε τους χρόνους που χρειάστηκαν για να ολοκληρωθούν οι αλγόριθμοι

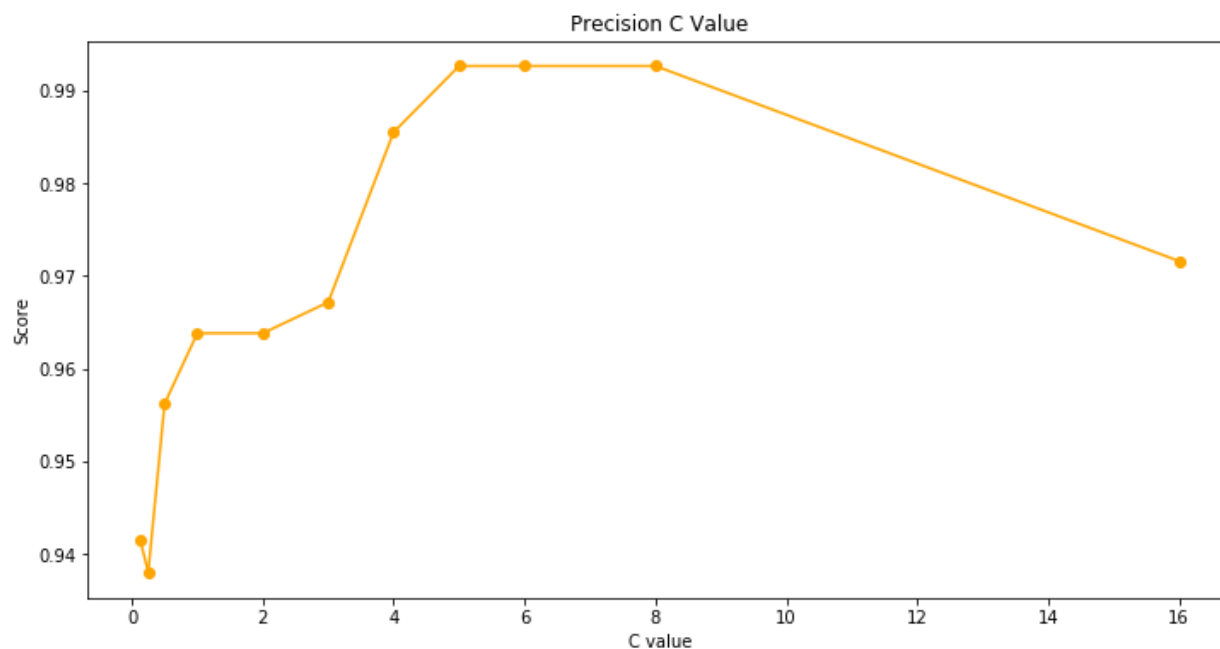
	0	1	2	3	4	5	6	7	8	9	10
Time Needed to finish	1.641953	1.526112	1.537131	1.542367	1.542747	1.490833	1.566025	1.555719	1.567346	1.550488	1.523901



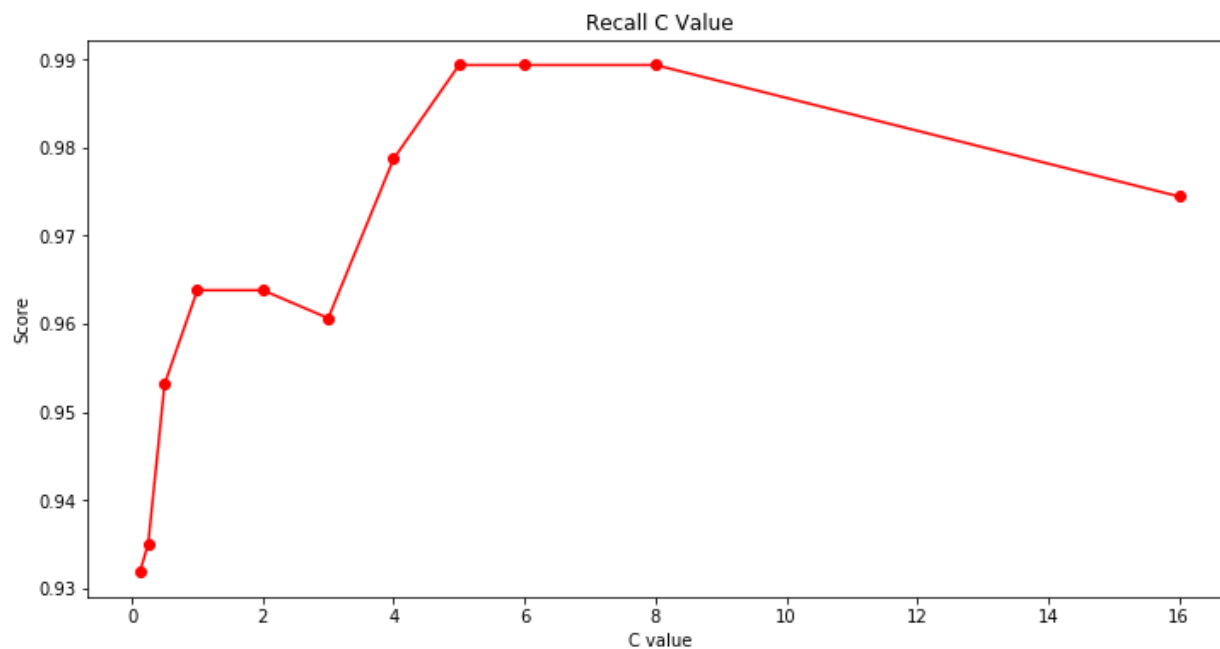
Ο γρηγορότερος όπως φαίνεται ήταν ο 5<sup>ος</sup> που αντιστοιχούσε για την τιμή του  $C = 2$

Ας δούμε τώρα τις μετρικές

## Precision

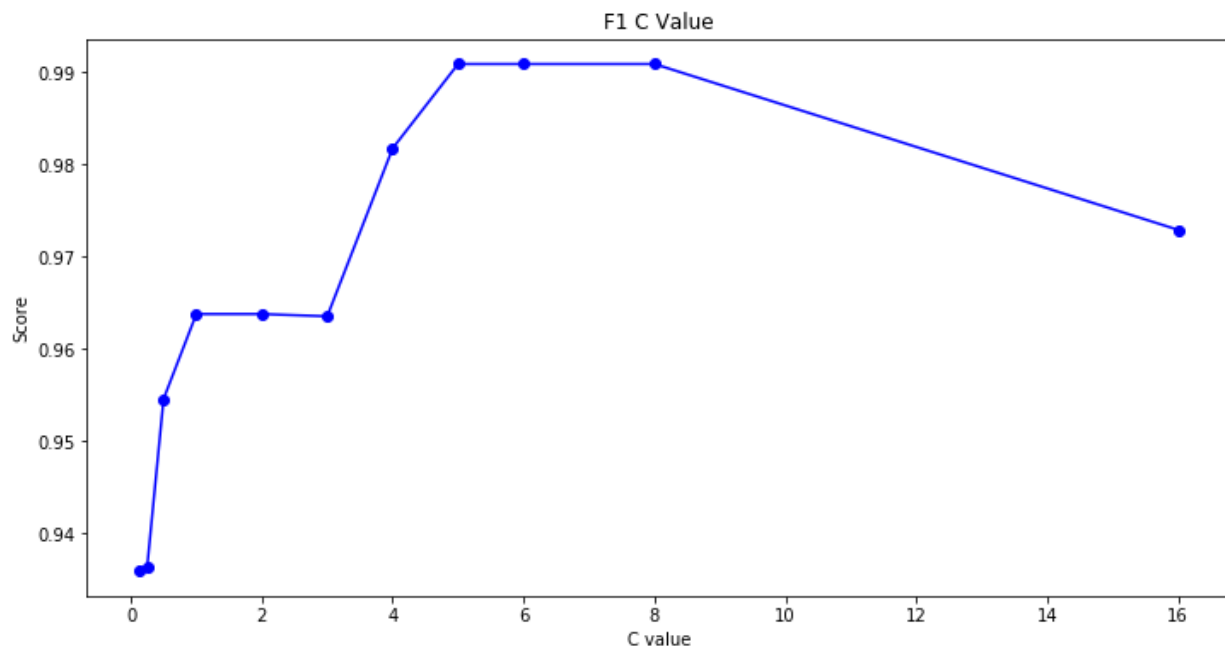


## Recall





## F1 Score



Οι καλύτερες επιδόσεις που βρέθηκαν αντιστοιχούσαν για την τιμή  $C=5$

### Classification Report.

	precision	recall	f1-score	support
-1.0	0.99	1.00	0.99	67
1.0	1.00	0.98	0.99	47
avg / total	0.99	0.99	0.99	114


Τα support vectors που χρειάστηκαν ήταν:

**102 support vectors out of 455 points**

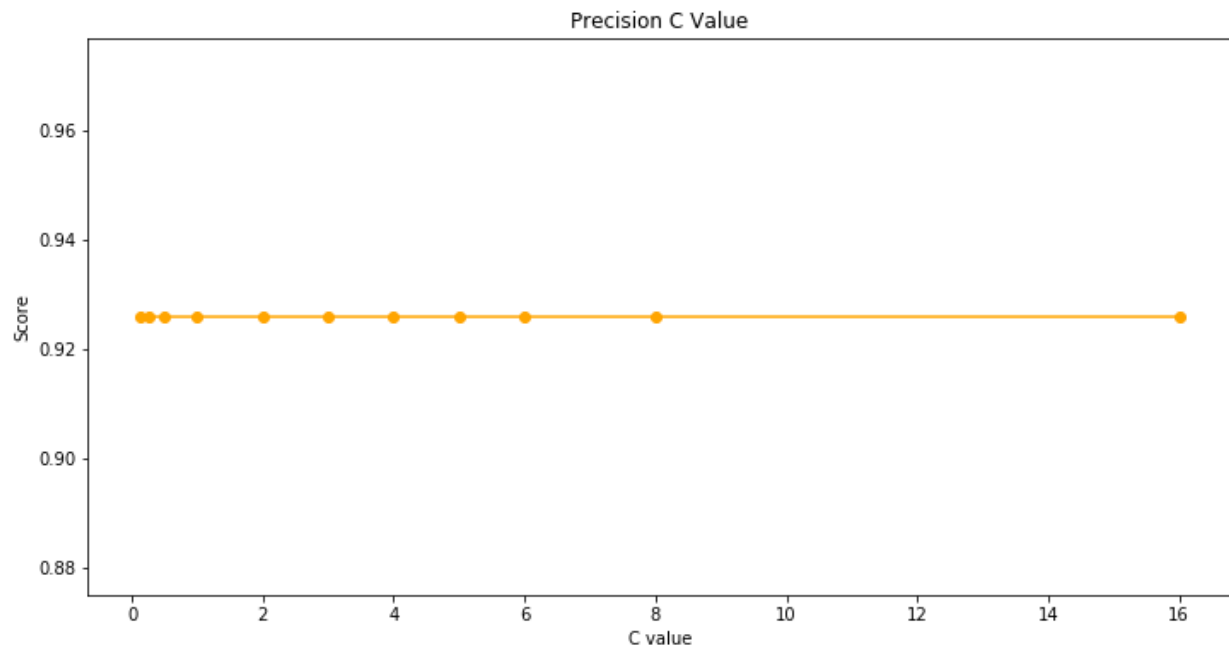
### Polynomial Kernel ( degree of polynomial $P=3$ )

Οι χρόνοι που χρειάστηκαν για να εκτελεστεί για καθένα από τα  $C$  ο αλγόριθμος SVM ήταν

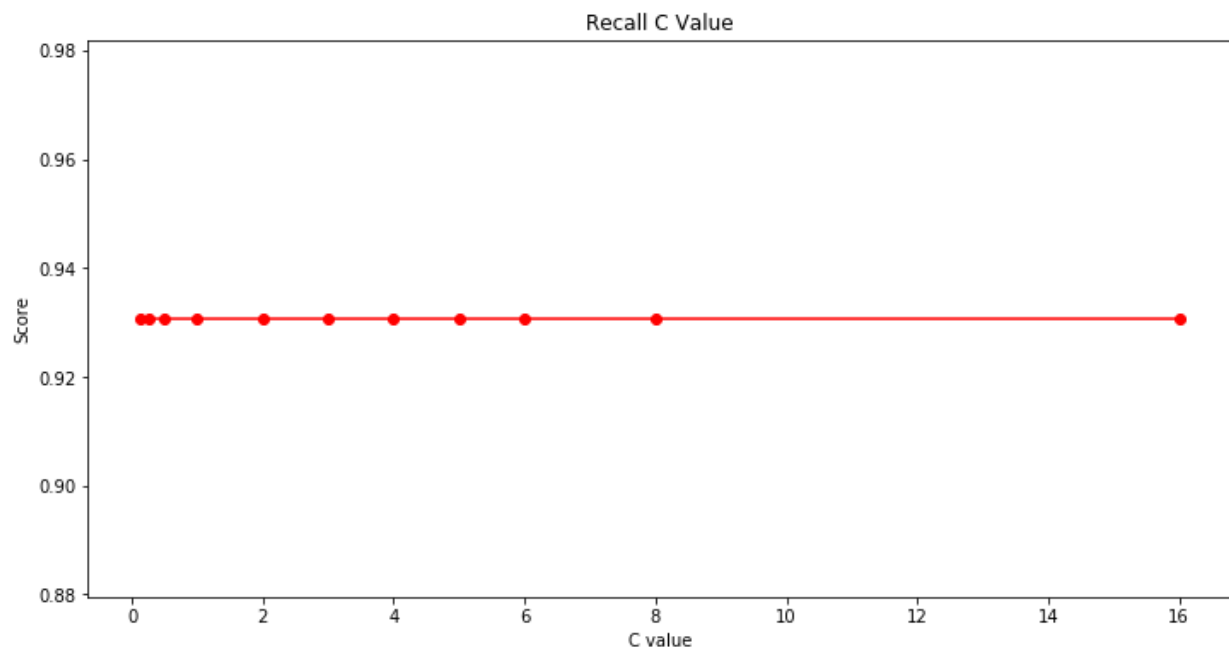
	0	1	2	3	4	5	6	7	8	9	10
Time Needed to finish	0.599997	0.621909	0.604569	0.615181	0.647058	0.627217	0.53557	0.633691	0.599735	0.645633	0.613371

Όπως βλέπουμε ο γρηγορότερος αλγόριθμος εκτελέστηκε  χρόνο **0.53557 seconds**

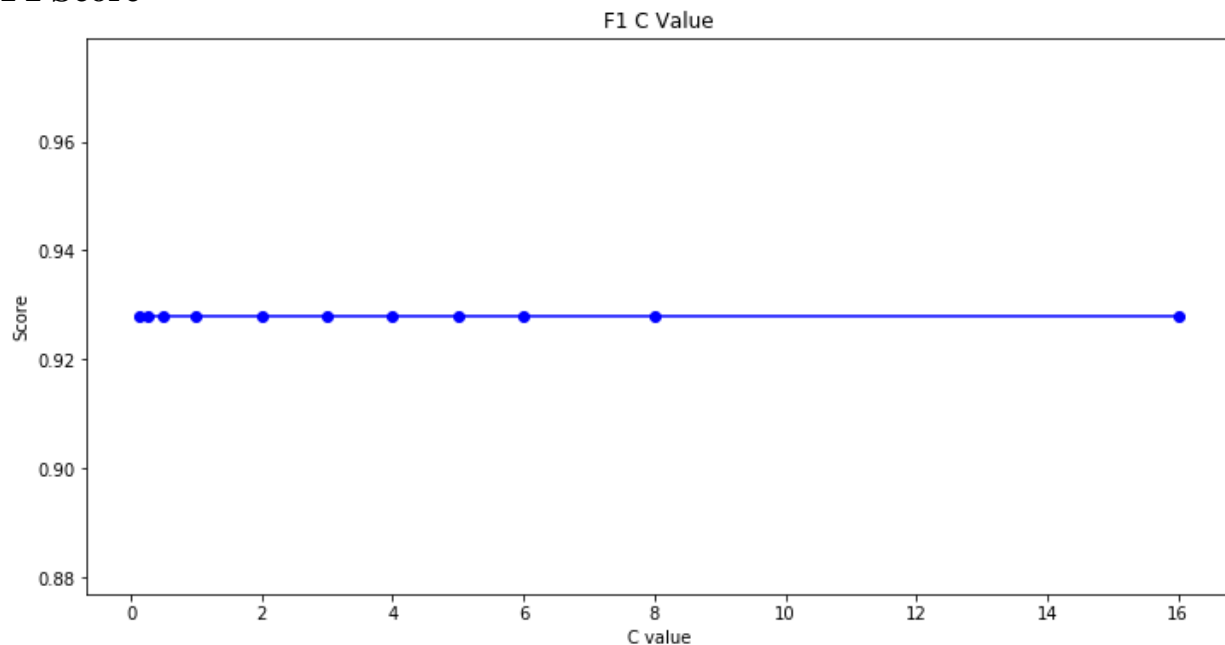
## Precision



## Recall



## F1 Score



Οι καλύτερες επιδόσεις βρέθηκαν για την τιμή του **C=0.125**

### Classification Report.

	precision	recall	f1-score	support
-1.0	0.95	0.93	0.94	67
1.0	0.90	0.94	0.92	47
avg / total	0.93	0.93	0.93	114

Τα support vectors που χρειάστηκαν ήταν:

**49 support vectors out of 455 points**

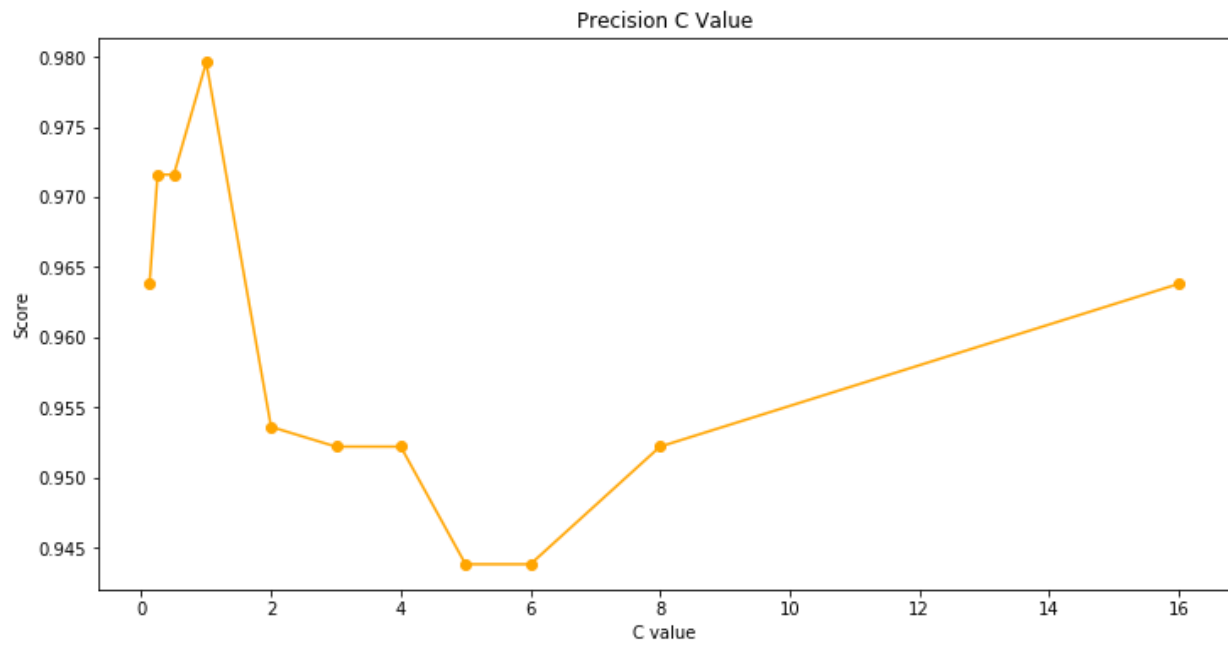
Ας δούμε τέλος τι θα συνέβαινε αν δεν είχαμε εφαρμόσει την διαδικασία **PCA**

### Linear Kernel

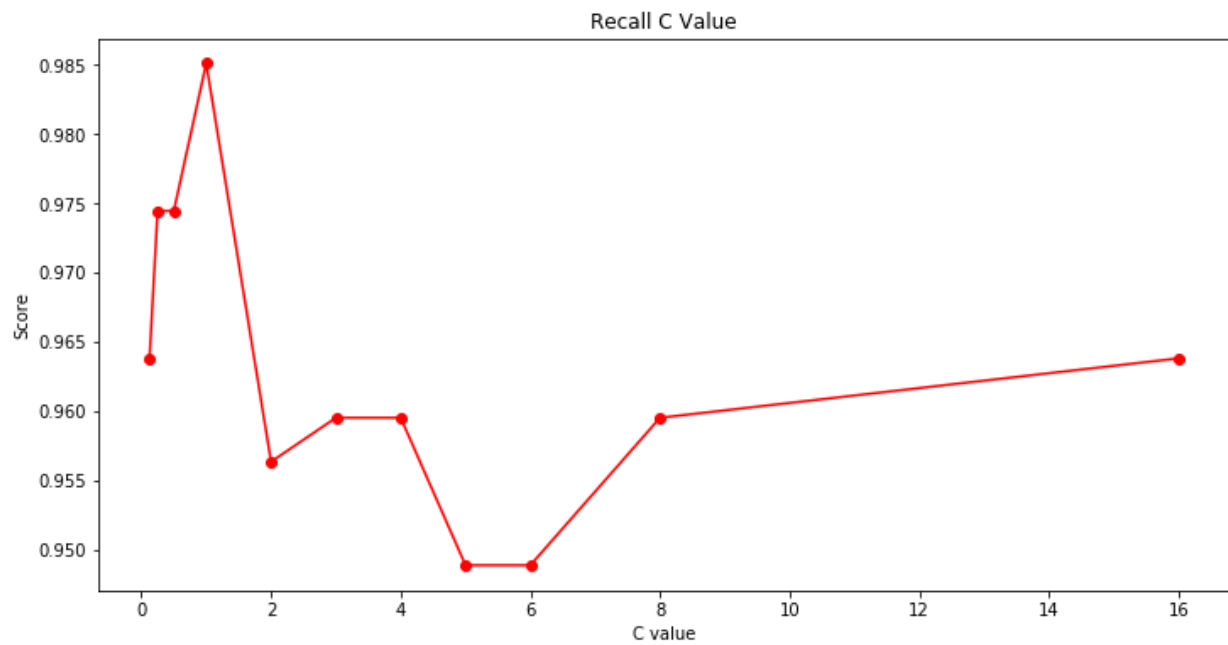
	0	1	2	3	4	5	6	7	8	9	10
Time Needed to finish	0.315807	0.423873	0.401697	0.401092	0.442557	0.402562	0.420142	0.413353	0.392701	0.422847	0.415123



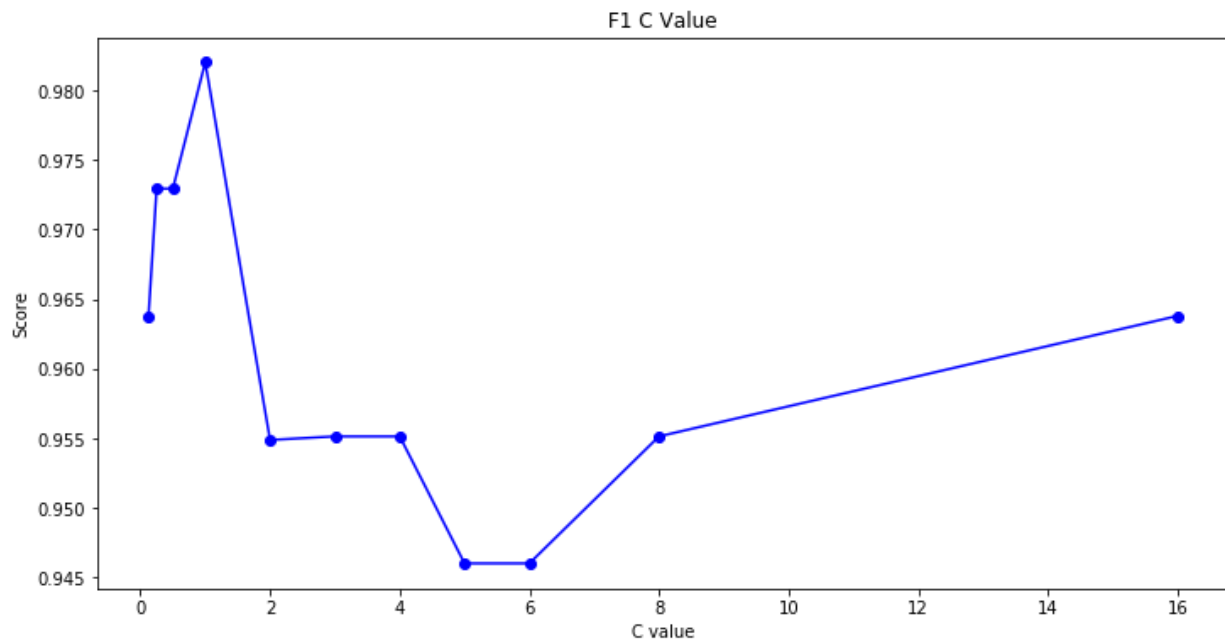
## Precision



## Recall



## F1 Score



Η μεγαλύτερη τιμή που βρέθηκε αντιστοιχούσε στην τιμή του  $C=1$

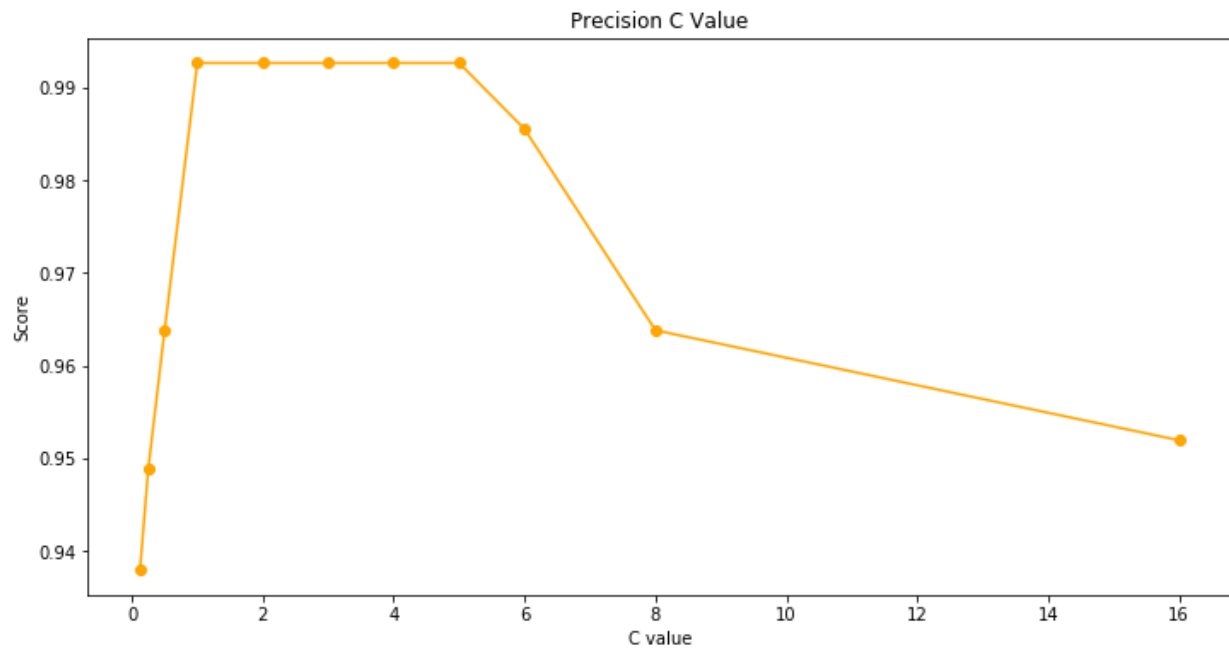
## Classification Report.

	precision	recall	f1-score	support
-1.0	1.00	0.97	0.98	67
1.0	0.96	1.00	0.98	47
avg / total	0.98	0.98	0.98	114

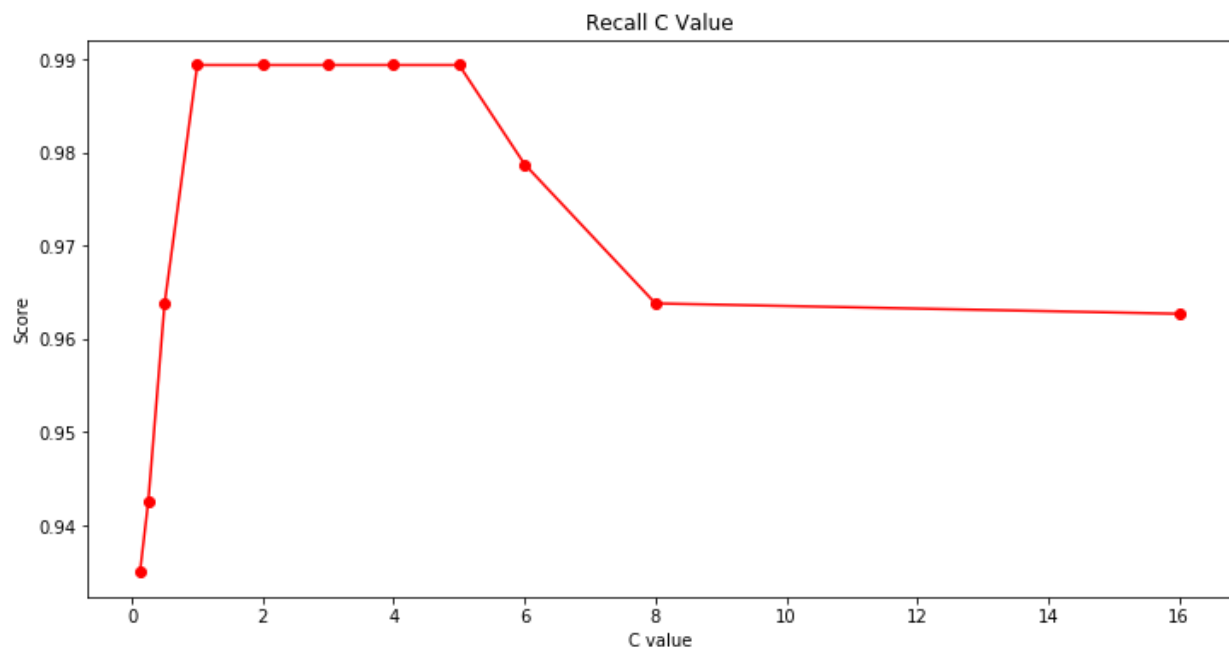
### Gaussian Kernel (RBF , sigma = 3)

[illegible]

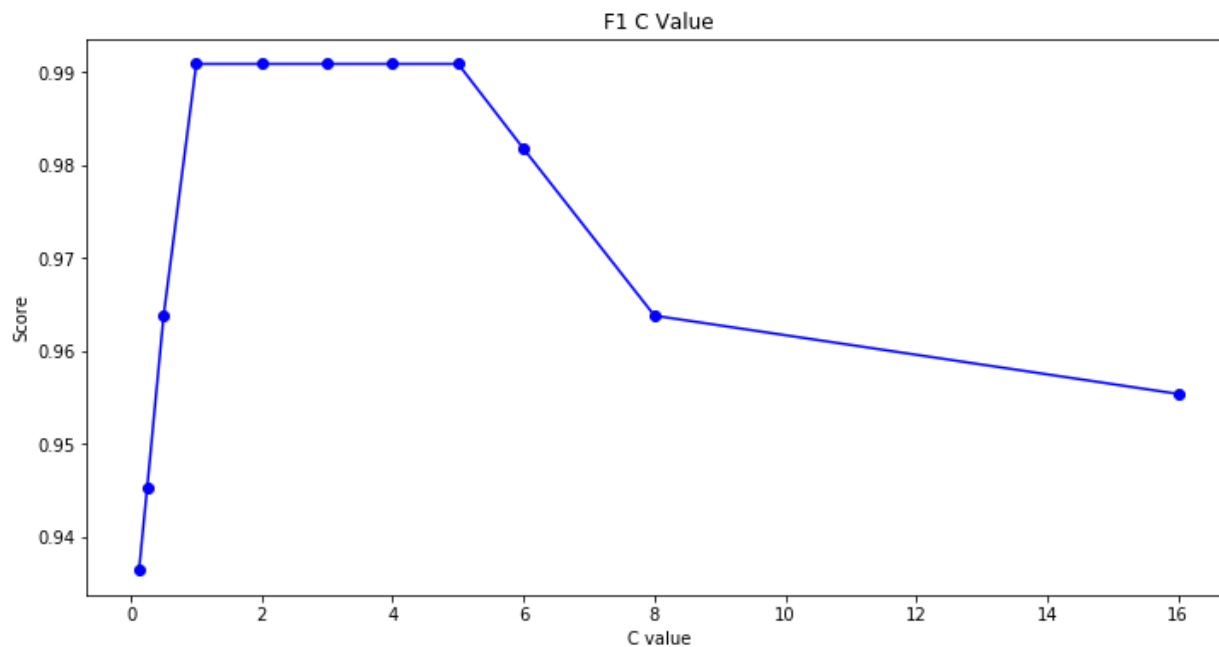
## Precision



## Recall



## F1 Score



Οι καλύτερες επιδόσεις βρέθηκαν για την τιμή του **C=1**

### Classification Report.

	precision	recall	f1-score	support
-1.0	0.99	1.00	0.99	67
1.0	1.00	0.98	0.99	47
avg / total	0.99	0.99	0.99	114

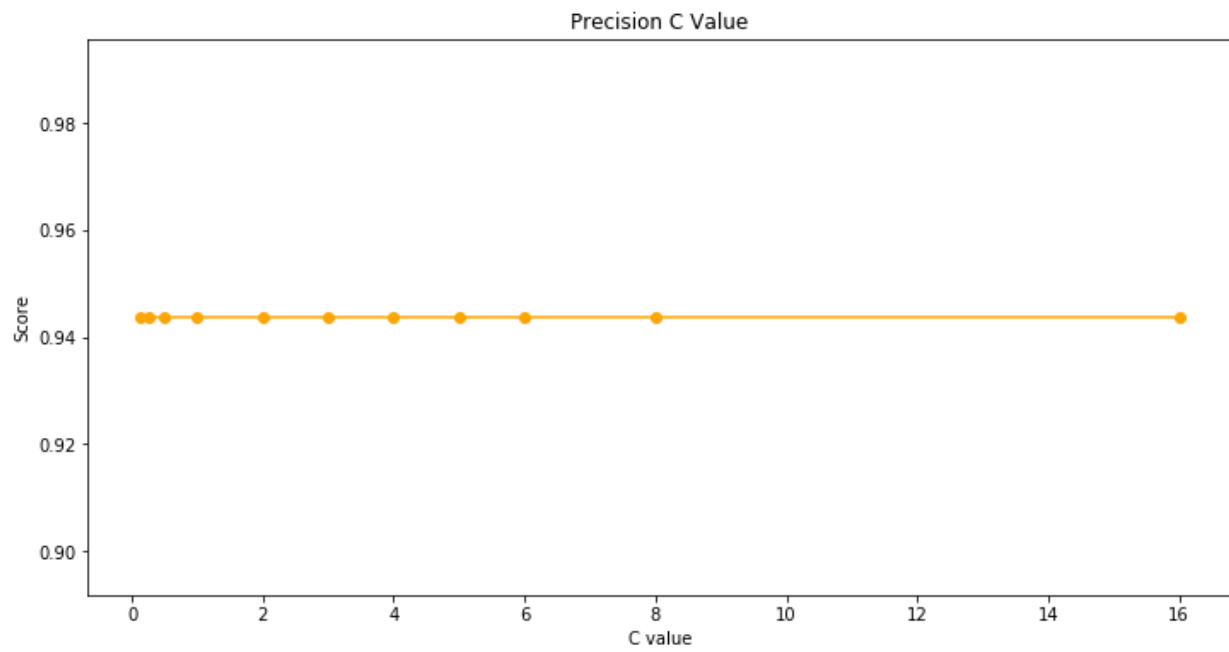
**133 support vectors out of 455 points**

### Polynomial Kernel ( degree of polynomial P=3 )

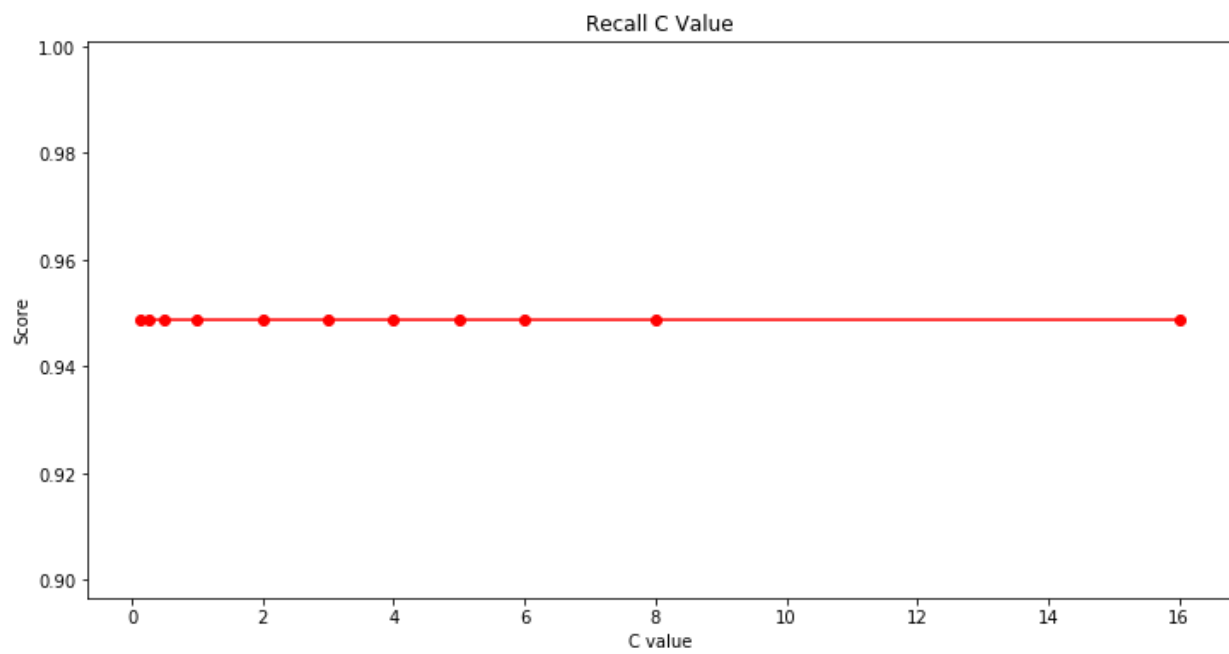
	0	1	2	3	4	5	6	7	8	9	10
Time Needed to finish	0.57518	0.558876	0.555224	0.584635	0.532551	0.535821	0.585132	0.554258	0.524061	0.574897	0.572772



## Precision

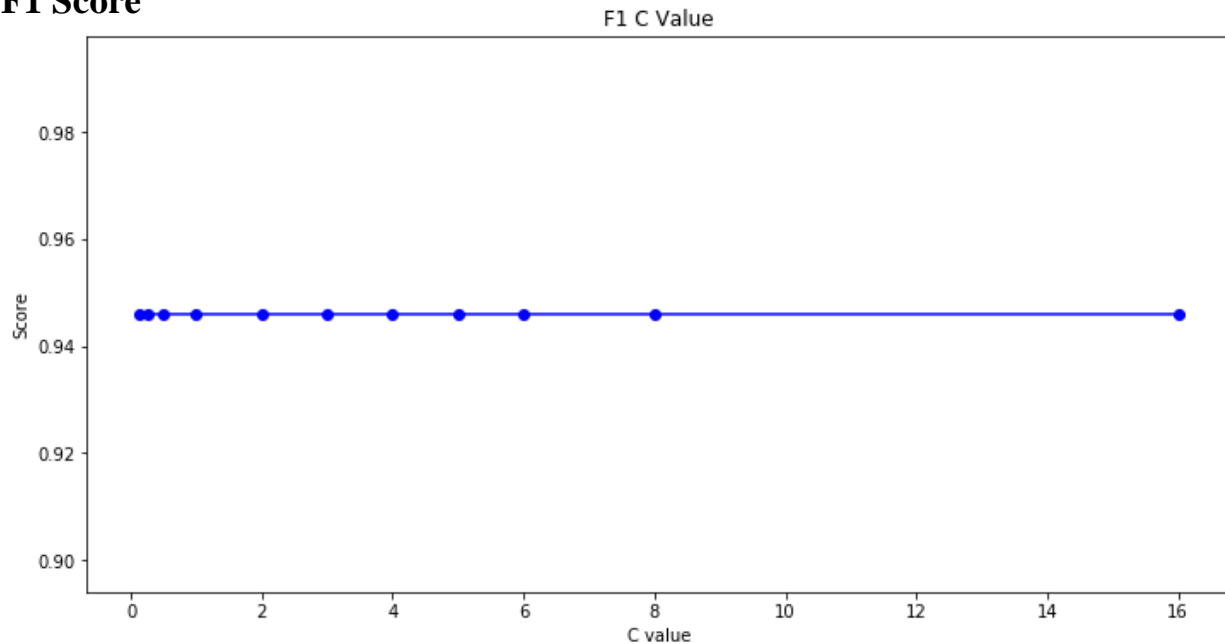


## Recall





## F1 Score



Οι μεγαλύτερες επιδόσεις βρέθηκαν για την τιμή του  $C=0.125$

## Classification Report.

	precision	recall	f1-score	support
-1.0	0.97	0.94	0.95	67
1.0	0.92	0.96	0.94	47
avg / total	0.95	0.95	0.95	114

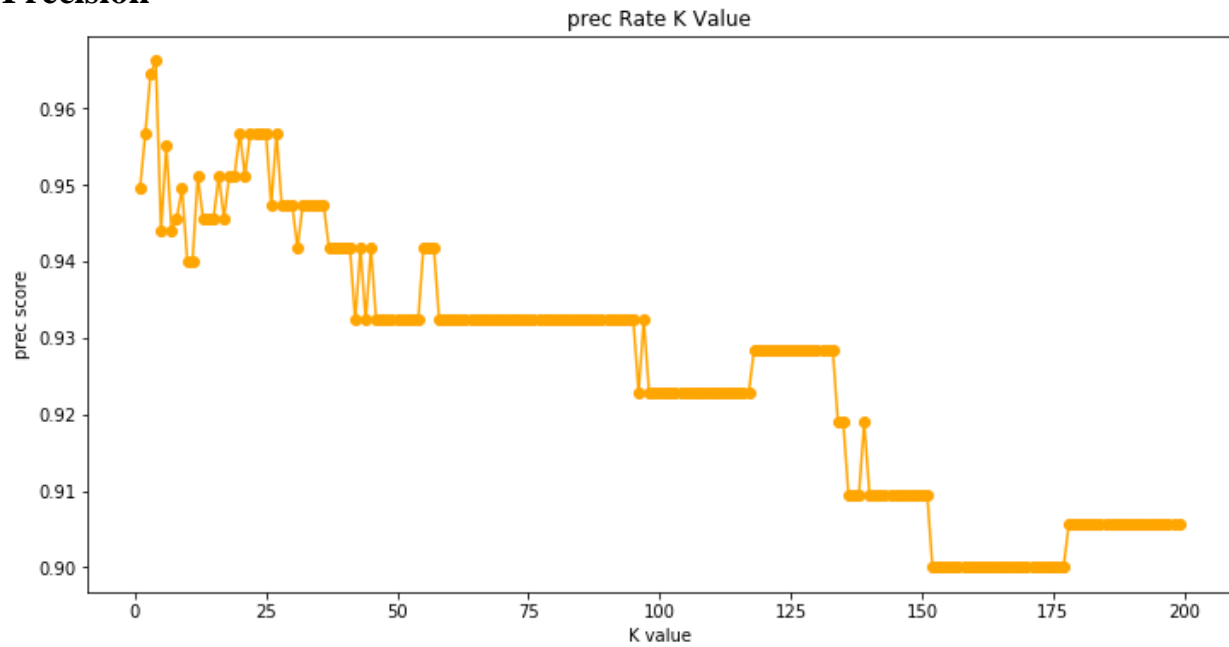
**64 support vectors out of 455 points**

- Σύγκριση με τους Αλγορίθμους **Nearest Neighbor** και **Nearest Class Centroid**

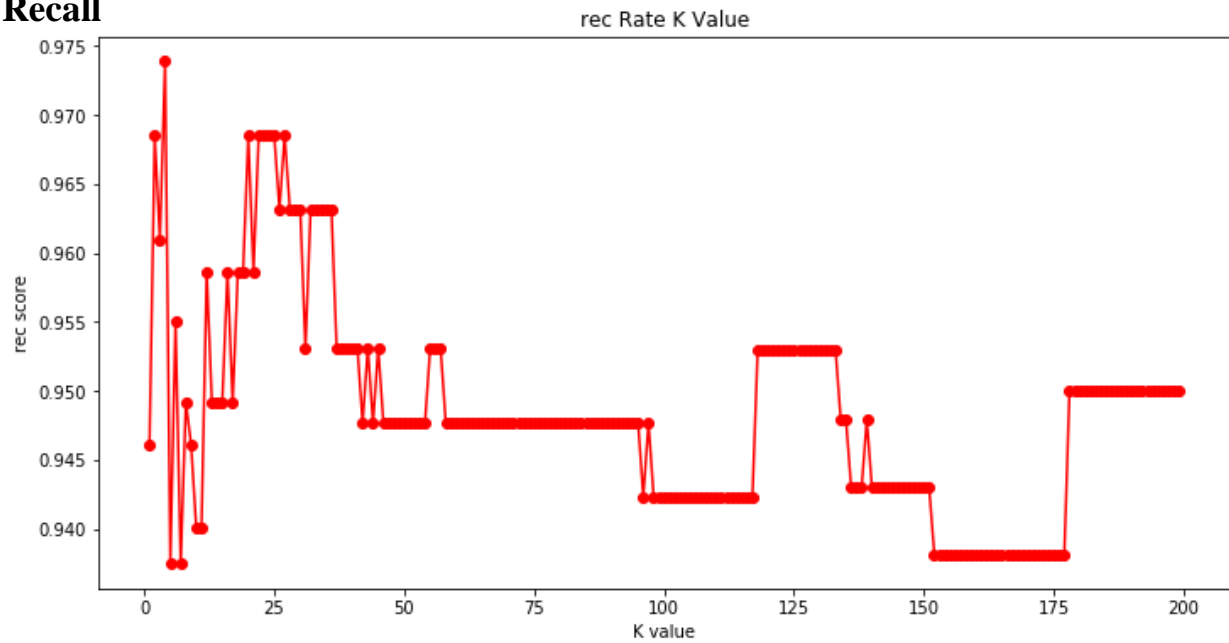
## Nearest Neighbor ( $K=200$ , distance="uniform", $p=2$ )

Τα αποτελέσματα τα οποία βρέθηκαν μετά την εφαρμογή του αλγορίθμου **K Nearest Neighbor (KNN)** ήταν:

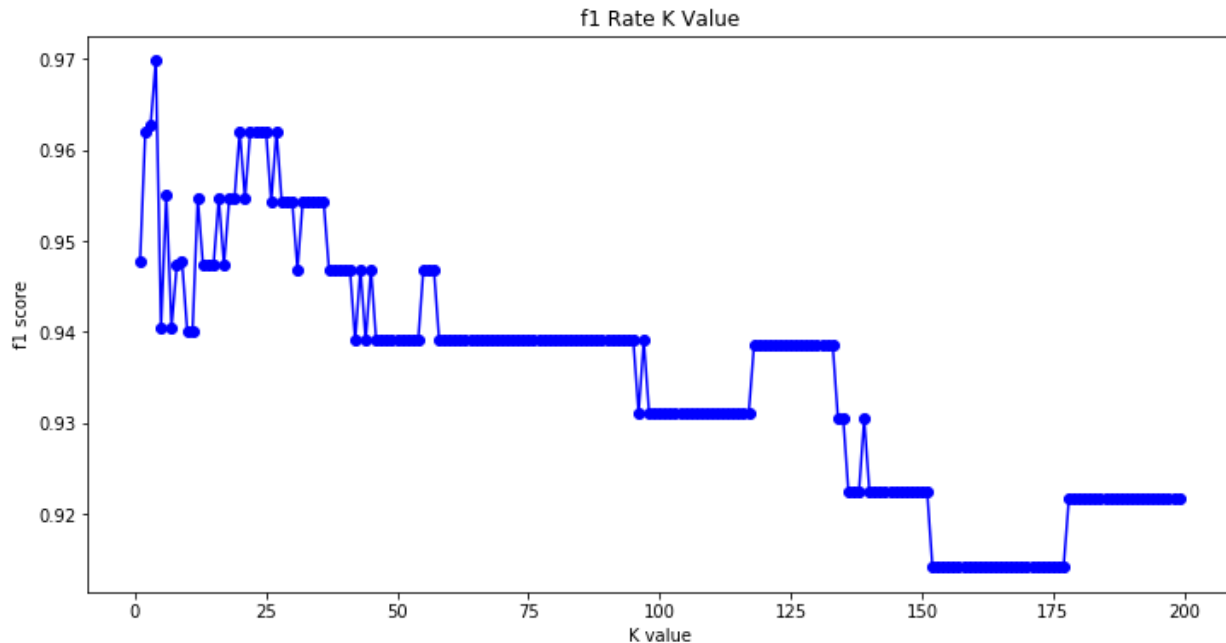
## Precision



## Recall



## F1 Score



Τα αποτελέσματα τα οποία συγκεντρώθηκαν ήταν:

Best **Precision** 0.9661425576519916 για την τιμή του **K=4**

Best **Recall** 0.9738917306052856 για την τιμή του **K=4**

Best **F1 Score** 0.9697802197802198 για την τιμή του **K=4**

**!Παρατηρήσεις:**

(Αν αντί του **standartscaler** εφαρμόζαμε **minmaxscaler** τότε τα αποτελέσματα που προέκυπταν ήταν)

Best **Precision** 0.9661425576519916 για την τιμή του **K=1**

Best **Recall** 0.9738917306052856 για την τιμή του **K=1**

Best **F1 Score** 0.9697802197802198 για την τιμή του **K=1**

Σε αυτήν την περίπτωση λοιπόν χρειάστηκε μόνο ένας πλησιέστερος γείτονας για να βρεί τα καλύτερα αποτελέσματα ο αλγόριθμος.

## Nearest Class Centroid

Με την εφαρμογή του MinMaxScaler τα αποτελέσματα που βρέθηκαν ήταν:

	precision	recall	f1-score	support
-1	0.94	0.93	0.94	90
1	0.89	0.91	0.90	53
avg / total	0.92	0.92	0.92	143

Ενώ με την εφαρμογή του StandardScaler τα αποτελέσματα που βρέθηκαν ήταν

	precision	recall	f1-score	support
-1	0.94	0.93	0.94	90
1	0.89	0.91	0.90	53
avg / total	0.92	0.92	0.92	143

Παρατηρούμε ότι βρήκαμε ακριβώς τα ίδια αποτελέσματα.

### ➤ Συγκρίσεις

Ας συγκεντρώσουμε λοιπόν τώρα όλα τα αποτελέσματα για να αποφανθούμε ποιος ητανε τελικά ο ακριβέστερος αλγόριθμος.

Είδαμε ότι στην περίπτωση του SVM τα καλύτερα αποτελέσματα βρέθηκαν στην περίπτωση του Gaussian Kernel (With and without pca)

Συγκεκριμένα: (Without PCA sigma=3 , C=1)

		precision	recall	f1-score	support
1. Best Precision : 0.9926470588235294.	-1.0	0.99	1.00	0.99	67
2. Best Recall : 0.9893617021276595	1.0	1.00	0.98	0.99	47
3. Best F1 Score : 0.9909199522102747	avg / total	0.99	0.99	0.99	114

Ακόμη: (With PCA sigma=3, C=5)

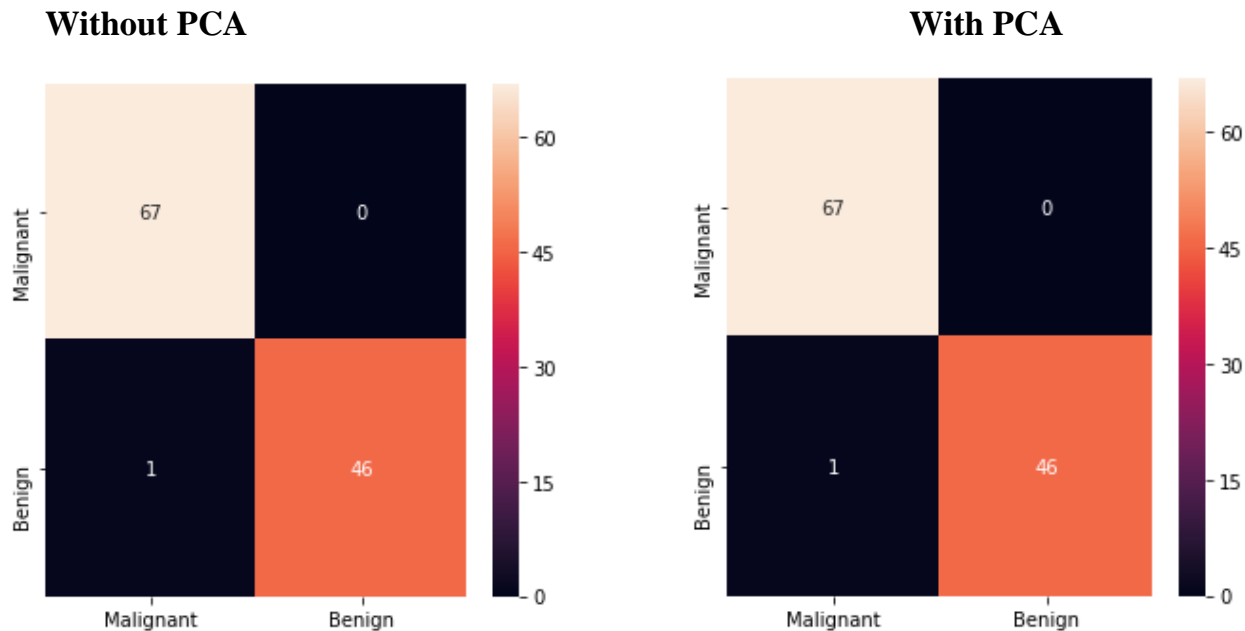
		precision	recall	f1-score	support
1. Best Precision: 0.9926470588235294	-1.0	0.99	1.00	0.99	67
	1.0	1.00	0.98	0.99	47
	avg / total	0.99	0.99	0.99	114

2. **Best Recall: 0.9893617021276595**

3. **Best F1 Score: 0.9909199522102747**

Ακόμη παραθέτουμε τους confusion matrix οι οποίοι προέκυψαν μετά την εκτέλεση του SVM

Σχήμα 1.11



Άρα πράγματι ο αλγόριθμος SVM είναι σε θέση να κατηγοριοποιεί με precision 99% την κλάση Benign και με ακρίβεια 100% την κλάση Malignant.

Στην περίπτωση του Nearest Neighbor τα καλύτερα αποτελέσματα βρέθηκαν για την περίπτωση **K=4** δηλαδή για 4 πλησιέστερους γείτονες και οι αποδόσεις ήτανε

1. **Precision = 0.9661425576519916**

2. **Recall = 0.9738917306052856**

3. **F1 Score = 0.9697802197802198**

Τέλος στην περίπτωση του Nearest Class Centroid τα αποτελέσματα ήτανε χαμηλότερα και από τους 2 αλγορίθμους της τάξης του:

1. **Precision = 92%**

2. **Recall = 92%**

3. **F1 Score = 92%**