Comparazione di classificatori in Credit Card Approval

Valerio Colitta, Daniele Cominu, Alessio Fiorenza

Aprile 2017

1. Descrizione preliminare

Volendo approfondire i metodi e le tecniche di classificazione affrontate nel corso di Metodi Quantitativi per l'Informatica, abbiamo deciso di confrontarli risolvendo il problema presentato in *Credit Approval Data Set* [1]; abbiamo deciso di utilizzare modelli di diverse complessità ed abbiamo osservato che data la conformazione dei dati e la loro dimensione ridotta, si ottengono risultati migliori con modelli meno complessi.

2. Il dataset

Il dataset riguarda delle domande di approvazione di carte di credito, ed e' composto da 690 campioni, 37 dei quali con valori mancanti. Sono presenti in totale 15 feature, le quali si articolano in 6 continue e 9 categoriche. Tali dati sono poi classificati in due classi {+, -} che rappresentano rispettivamente l'approvazione o meno della carta di credito. Il significato delle 15 feature non è noto per poter mantenere la confidenzialità dei dati.

3. Manipolazione dei dati

Come prima cosa sono stati eliminati manualmente dal dataset i 37 campioni di cui non erano stati specificate alcune feature, riducendo ulteriormente la dimensione del dataset a 653 campioni.

3.1. One Hot Encoding

Per superare l'eterogeneità nella tipologia delle feature, si è deciso di applicare la tecnica del *One Hot Encoding* [2] per le feature categoriche; l'*One Hot Encoding* è una tecnica utilizzata per trattare feature categoriche in problemi di classificazione e regressione, e consiste nel tradurre una feature categorica che può assumere n valori distinti in un vettore di n feature binarie; per ogni campione la feature i-esima del vettore calcolato assume il valore 1 se e solo se il campione assume il valore i-esimo per la feature considerata; le restanti n - 1 sono dunque settate a 0 per tale campione. Se si fosse utilizzata una feature a valori reali, in cui per ogni valore assunto dalla feature viene associato un numero reale, sarebbero state introdotte delle distanze diverse tra i valori assunti dalla feature; Ad esempio, considerando una feature categorica che descrive la specie di appartenenza, mappando i valori assunti a dei numeri incrementali, si può notare come si introduca una maggiore distanza tra i campioni (2, 7) rispetto ai campioni (4, 5).

Sample	Category	Numerical
1	Human	1
2	Human	1
3	Penguin	2
4	Octopus	3
5	Alien	4
6	Octopus	3
7	Alien	4

Figura 1: mapping categorica -> reale

Introducendo il one hot encoding si può notare come le distanze tra i campioni (2,7) e (4,5) siano le stesse

Sample	Human	Penguin	Octopus	Alien
1	1	0	0	0
2	1	0	0	0
3	0	1	0	0
4	0	0	1	0
5	0	0	0	1
6	0	0	1	0
7	0	0	0	1

Figura 2: one hot encoding

3.2. Standardizzazione

Successivamente all'one hot encoding, è seguita la fase di standardizzazione delle variabili; per ogni feature sono stati calcolati la media campionaria μ e la deviazione standard campionaria σ, definite come:

$$\mu_j = \frac{1}{N}\sum_{i=1}^{N} x_{ij} \tag{1}$$

$$\sigma_j = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_{ij}-\mu_j)^2} \tag{2}$$

Una volta calcolati questi due valori, i campioni vengono standardizzati attraverso la seguente formula:

$$x_{ij} = \frac{x_{ij}-\mu_j}{\sigma_j} \tag{3}$$

Questa trasformazione fa si che le nuove feature abbiano valor medio μ nullo e varianza σ^2 unitaria; ciò è necessario poiché alcuni metodi utilizzano la distanza euclidea e potrebbero essere ingannati dall'utilizzo di diverse scale per le diverse feature.

4. Principal Component Analysis

La Principal Component Analysis è una tecnica di dimensionality reduction, ossia viene utilizzata per proiettare i campioni su un universo con un numero minore di dimensioni rispetto all'originale; viene utilizzata per diversi scopi tra cui la visualizzazione dei dati, per cogliere le dimensioni principali su cui sono distribuiti i dati (*latent factors*) e per ridurre il volume dei dati ed in tal modo il tempo computazionale degli algoritmi di learning.

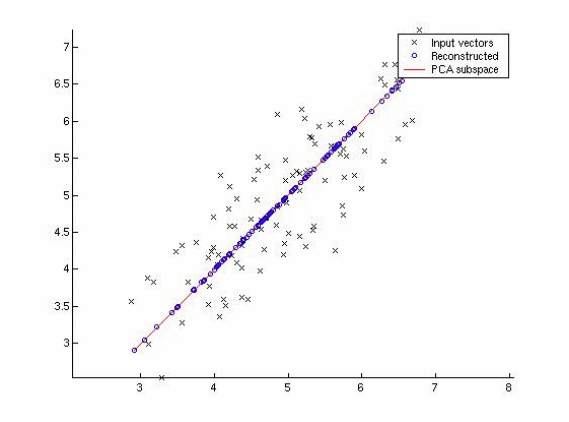

Figura 3: PCA

Con la PCA vogliamo trovare una matrice di trasformazione W tale che possiamo approssimare i campioni del dataset $x_i \in \mathbb{R}^D$ con dei vettori $\hat{x_i}$ ottenuti da $\hat{x_i} = Wz_i$, ove $z_i \in \mathbb{R}^L$ con $L < D$ e W matrice ortogonale DxL. Vogliamo quindi minimizzare la distanza tra i campioni originali x_i e le nostre approssimazioni $\hat{x_i}$, ossia matematicamente il *reconstruction error*

$$\begin{aligned} J(W,Z) &= \frac{1}{N}\sum_{i=1}^{N}||x_i - Wz_i||^2 \\ &= ||X^T - WZ^T||_F^2 \end{aligned} \tag{4}$$

dove $||A||_F$ identifica la *Frobenius Norm*, ossia la radice della somma dei quadrati di tutti gli elementi della matrice.
Si osserva che la matrice W che minimizza la norma di Frobenius è la matrice V_L nella Singular Value Decomposition (SVD); infatti effettuando la SVD sulla Design Matrix X, si ottiene

$$\begin{aligned} &X = USV^T \implies X_L = U_L S_L V_L^T = ZW^T \\ &ove \quad U_L^T U_L = I_L, \quad V_L^T V_L = I_D \end{aligned} \tag{5}$$

con S_L è la matrice diagonale contenente gli autovalori in ordine decrescente ottenuta da S ponendo a 0 le ultime $D-L$ componenti sulla diagonale. Infatti il *reconstruction error* risulta pari a la somma dei $D-L$ autovalori esclusi:

$$\begin{aligned} ||X - X_L||_F &= ||U(S - \begin{bmatrix} S_L & \\ & 0 \end{bmatrix})V^T||_F \\ &= ||S - \begin{bmatrix} S_L & \\ & 0 \end{bmatrix}||_F \\ &= \sum_{i=L+1}^{D}\sigma_i \end{aligned} \tag{6}$$

5. Visualizzazione dei dati

In genere risulta estremamente utile effettuare una visualizzazione dei dati prima di procedere all'applicazione dei modelli, in modo da individuare pattern utili per guidare il processo di selezione del modello e l'analisi dei risultati ottenuti. I grafici in Figura 4 rappresentano le distribuzioni dei campioni considerando coppie di feature continue; ossia nella cella presente alla riga *i-esima* e colonna *j-esima* è presente un grafico in cui sull'asse x sono riportati i valori assunti dalla feature *j-esima*, sull'asse y quelli assunti dalla feature *i-esima* .

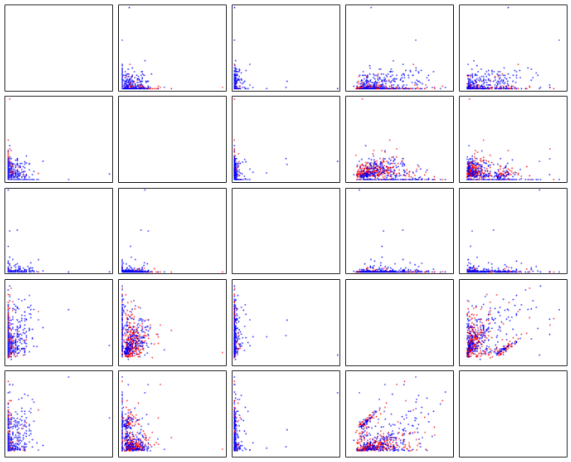

Figura 4

Dai grafici non si evincono particolari legami tra le feature. E' pertanto necessario continuare ad investigare.

6. Modelli utilizzati

Si è deciso di affrontare il problema utilizzando diversi modelli, in modo da poter approfondire la struttura del problema proposto. Si è notato come modelli più semplici riescano a generalizzare maggiormente a fronte di nuovi dati di input, e ciò è dovuto a diversi fattori, tra cui la dimensione ridotta del dataset e il numero relativamente alto di feature. Prima del training il dataset è stato inizialmente diviso in due parti: *Training set* (80%) e *Test set* (20%), utilizzati rispettivamente per il training e la selezione degli iperparametri attraverso la cross-validation e il test dei risultati ottenuti.
Nella Tabella 1 sono riportati gli error-rate medi e minimi ottenuti applicando i diversi modelli.

Modello	Min	Med
Linear Discriminant Analysis	0.1021	0.1167
Quadratic Discriminant Analysis	0.1094	0.1167
Diagonal Discriminant Analysis	0.0948	0.1021
Logistic Regression (LB)	0.0948	0.1167
Logistic Regression (QB)	0.1459	0.1970
Logistic Regression (LBR)	0.0948	0.1094
Logistic Regression (QBR)	0.1240	0.1386

Tabella 1: errori minimi e medi dei modelli utilizzati

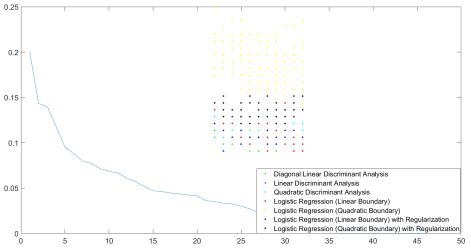

Figura 5: Error Rate dei vari modelli

7. Gaussian Discriminant Analysis

La Gaussian Discriminant Analysis, o GDA, è una tecnica di classificazione di tipo *generativa*, e dunque calcola la distribuzione $p(x|y,\theta)$ (la class conditional density) di una serie di campioni, a partire dalla loro classe y di appartenenza. Con la GDA, si assume che i campioni di ogni classe c siano distribuiti come delle Gaussiane Multivariate, nel senso che

$$p(x|y=c) \sim \mathcal{N}(\mu_c, \Sigma_c) \quad \forall c \in \{1,...,C\} \qquad (7)$$

dove μ_c rappresenta il vettore dei valori medi della distribuzione, e Σ_c la matrice di covarianza delle feature. Una volta trovati i due parametri (μ, Σ), per classificare un nuovo input $\hat{x}$ in una delle C classi viene calcolato

$$\arg\max_c p(y=c|x=\hat{x}) \quad \forall c \in \{1,...,C\} \qquad (8)$$

e cioè la massima probabilità che la classe di appartenenza di $\hat{x}$ sia c, per tutte le C classi.
Si osserva che indicando il prior con π, $\hat{y}(x)$ risulta:

$$\arg\max_c [\log p(y=c|\pi_c) + \log p(\hat{x}|y=c,\theta_c)] \qquad (9)$$

$$\arg\min_c [-\log(\pi_c) + \frac{1}{2}\log|\Sigma_c| + \frac{1}{2}(\hat{x}-\mu_c)^T\Sigma_c^{-1}(\hat{x}-\mu_c)] \qquad (10)$$

L'ultimo parametro è la distanza di Mahalanobis al quadrato dal valore medio μ_c della classe

c, ossia una distanza pesata con l'inverso degli autovalori di Σ_c lungo i suoi autovettori:

$$\begin{aligned}\|x-\mu_c\|_\Sigma^2 &= (\hat{x}-\mu)^T\Sigma^{-1}(\hat{x}-\mu)\\ &= (\hat{x}-\mu)^T(\sum_{i=1}^{D}\frac{1}{\lambda_i}u_iu_i^T)(\hat{x}-\mu)\\ &= \sum_{i=1}^{D}\frac{1}{\lambda_i}(\hat{x}-\mu)^Tu_iu_i^T(\hat{x}-\mu) \qquad (11)\\ &= \sum_{i=1}^{D}\frac{y_i^2}{\lambda_i}\\ ove \quad y_i &= \mu_i^T(x-\mu)\end{aligned}$$

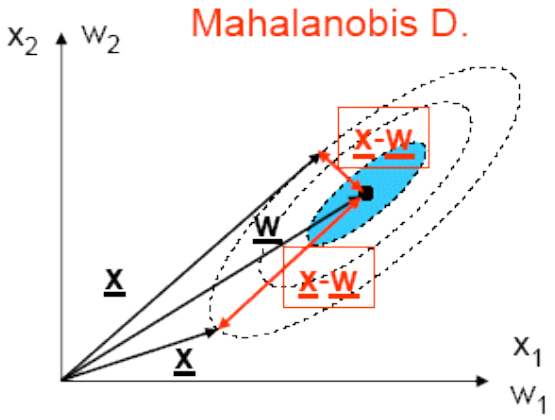

Figura 6: distanza di Mahalanobis

7.1. Quadratic Discriminant Analysis

Ne consegue che il decision boundary tra due classi $(0,1)$ essendo definito come :

$$p(y=0|x) = p(y=1|x) \qquad (12)$$

corrisponde a :

$$(x-\mu_0)^T\Sigma_0^{-1}(x-\mu_0) = (x-\mu_1)^T\Sigma_1^{-1}(x-\mu_1) + c \qquad (13)$$

Avendo un'equazione quadratica in entrambi i termini, nel caso generico il decision boundary si presenta come un paraboloide.

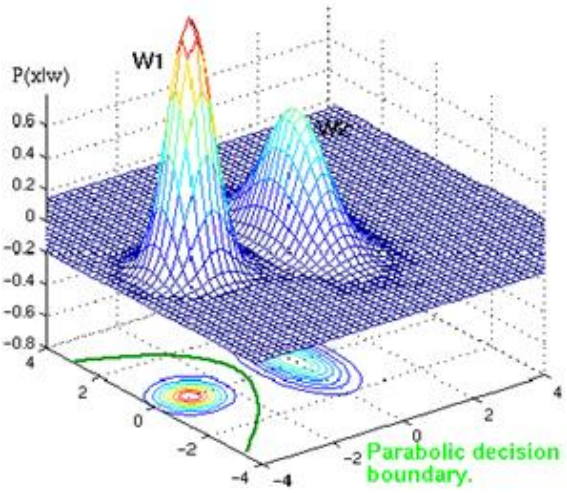

Figura 7: Decision Boundary QDA

7.2. Linear Discriminant Analysis

Assumendo che

$$\Sigma_c = \Sigma \quad \forall\, c = \{1,...,C\} \qquad (14)$$

$$\begin{aligned}p(y=c|x,\theta) &\propto \pi_c e^{-\frac{1}{2}(x-\mu_c)^T\Sigma^{-1}(x-\mu_c)}\\ &= e^{-\frac{1}{2}x^T\Sigma^{-1}x}e^{\mu_c^T\Sigma^{-1}x-\frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c+\log\pi_c} \qquad (15)\\ &\propto e^{-\frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c+\log\pi_c}\end{aligned}$$

dove il primo termine è stato rimosso poiché non dipende dalla classe di appartenenza; poiché l'unico termine quadratico è stato rimosso, si possono evidenziare i termini γ_c e β_c

$$\begin{aligned}\gamma_c &= -\frac{1}{2}\mu_c^T\Sigma^{-1}\mu_c + \log\pi_c\\ \beta_c &= \Sigma^{-1}\mu_c\end{aligned} \qquad (16)$$

ottenendo

$$p(y=c|x,\theta) \propto e^{\beta_c^Tx+\gamma_c} \qquad (17)$$

da cui si può evincere che il decision boundary è lineare

$$\begin{aligned}p(y=c|x,&\theta) = p(y=c'|x,\theta)\\ &\Longleftrightarrow\\ \beta_c^Tx+\gamma_c &= \beta_c'^Tx+\gamma_c' \qquad (18)\\ &\Longleftrightarrow\\ (\beta_c-\beta_{c'})^Tx &= -(\gamma_c+\gamma_{c'})\end{aligned}$$

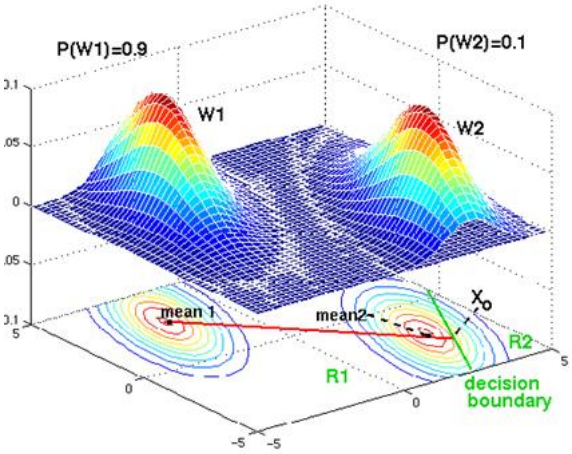

Figura 8: Decision Boundary LDA

7.3. Diagonal Linear Discriminant Analysis

Consideriamo il caso in cui $\Sigma_c = \Sigma = diag(\sigma_1^2, ..., \sigma_D^2)$ per ogni classe. In tal caso gli autovettori di Σ sono paralleli agli assi, e da quanto detto prima si deduce che $\hat{y}$ risulta pari a

$$\begin{aligned} &\arg\min_c[-\log(\pi_c) + \frac{1}{2}(\hat{x} - \mu_c)^T\Sigma^{-1}(\hat{x} - \mu_c)] \\ &= \arg\min_c[-\log(\pi_c) + \frac{1}{2}\sum_{i=1}^{D}\frac{(x_i - \mu_{ci})^2}{\sigma_i}] \end{aligned} \tag{19}$$

La Diagonal Linear Discriminant Analysis avendo meno parametri da stimare si comporta meglio in presenza di campioni con un alto numero di feature e dimensione del dataset ridotte, in quando tende a fare un minore overfitting.

8. Logistic Regression

La Logistic Regression è una tecnica di classificazione binaria di tipo *discriminativo*, ossia si stima direttamente la $p(y|x)$ senza passare per class conditional density; in particolare si ottiene dalla *Linear Regression* considerando le y distribuite come una Bernulliana di parametro $\mu(x) = sigm(w^T x)$.

$$y \sim Ber(y|\mu(x)), \quad \mu(x) = sigm(w^T x) \tag{20}$$

Differentemente dalla Linear Regression non è possibile trovare analiticamente con una formula chiusa il valore di w che minimizza p(D|w), e dunque è necessario utilizzare degli algoritmi di minimizzazione. Calcolando la negative log-likelihood si ottiene infatti:

$$\begin{aligned} NLL &= -\log p(D|\theta) = \\ &= \sum_i [y_i log(sigm(w^T x)) + \\ &\quad + (1 - y_i) log(1 - sigm(w^T x))] \end{aligned} \tag{21}$$

8.1. Linear Boundary

Dato che la

$$\begin{aligned} &p(y|x, w) = Ber(y|sigm(w^T x)) \\ &= sigm(w^T x)^{I(y=1)}(1 - sigm(w^T x))^{I(y=0)} \end{aligned} \tag{22}$$

allora ponendo $p(y = 1|x, w) = p(y = 0|x, w)$ per trovare il decision boundary si ottiene

$$\begin{aligned} &sigm(w^T x) = (1 - sigm(w^T x)) \\ &\iff \\ &sigm(w^T x) = 0.5 \implies w^T x = 0 \end{aligned} \tag{23}$$

che dunque risulta essere lineare.

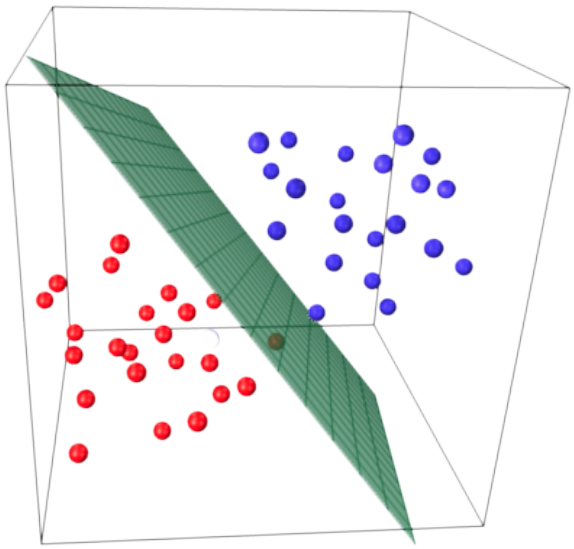

Figura 9: Linear Boundary Logistic Regression

8.2. Quadratic Boundary

Per ottenere un boundary non lineare si può sostituire ai campioni x un vettore $\phi(x)$ che contiene funzioni della x; in particolare con un ragionamento analogo a quello utilizzato nel caso lineare si osserva che il decision boundary sarà dato da

$$w^T\phi(x) = 0 \tag{24}$$

Dunque, utilizzando una **multivariate polynomial expansion** di grado 2, è possibile ottenere un boundary quadratico.

$$w^T\phi(x) = w_0 + \sum_{i=1}^{D} w_i x_i + \sum_{i,j:i<=j}^{D} w_{ij} x_i x_j \tag{25}$$

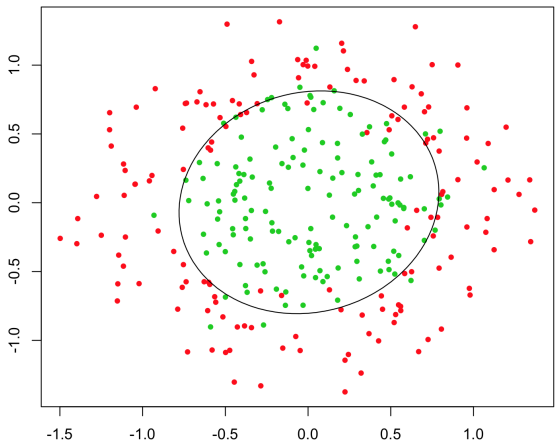

Figura 10: Quadratic Boundary Logistic Regression

8.3. Regolarizzazione

In presenza di un dataset ristretto e di molti parametri in un modello, ad esempio utilizzando un polinomio di grado alto, è possibile che i parametri trovati dal modello non generalizzino bene a fronte di nuovi dati di input, ossia è facile che il modello faccia overfitting sul training set; in questi casi i valori assunti da w sono specifici per i dati in ingresso e tendono a fittare anche il rumore, facendo assumere a w dei coefficienti elevati per poter seguire tutte le variazioni di quest'ultimo.
Un modo per limitare la flessibilità del modello rispetto al rumore, e in tal modo ridurre l'overfitting consiste nell'aggiungere dei pesi ai coefficienti w, in modo tale che durante la minimizzazione dell'errore, questi tendano a non assumere valori troppo elevati.
Tali pesi possono essere espressi come un prior su w, distribuito come una normale di centro 0 e varianza λ, in modo da poter influenzare la distribuzione di probabilità $p(w|D)$

$$p(w) = \mathcal{N}(0, \lambda I) \quad p(w|D) \propto p(D|w)p(w) \tag{26}$$

da cui la funzione da minimizzare diventa

$$f(w) = NLL(w) + \lambda w^T w \tag{27}$$

9. Bias vs Variance

Quando si parla di *prediction error*, questo può essere decomposto in tre componenti: errore dovuto al *bias*, errore dovuto alla *varianza* ed errore irriducibile dovuto al rumore. E' importante individuare quali sono le componenti d'errore nel nostro modello, per poter comprendere come questo si stia comportando e se stia effettuando under o over-fitting. Il *Bias* e la *Variance* sono definiti a livello matematico come:

$$\begin{aligned} Bias &= E[\hat{f}(x)] - f(x) \\ Variance &= E\left[\left(\hat{f}(x) - E[\hat{f}(x)]\right)^2\right] \end{aligned} \tag{28}$$

Intuitivamente il Bias rappresenta l'errore sulla media delle nostre predizioni per ogni campione del dataset, supponendo di effettuare il training del modello con diversi dataset, mentre la Variance rappresenta quanto variano le nostre predizioni tra di loro per un dato campione.
In genere vorremmo avere Bias e Variance nulli, ossia vorremmo predire sempre il valore corretto, ma ci si rende conto che esiste un trade-off tra le due fonti di errore, infatti si può intuire una correlazione tra Bias/Variance e Underfitting/Overfitting: più il nostro modello farà overfitting sul training set, più varieranno le predizioni al variare dei dati in input e dunque si avrà un'alta Variance; d'altra parte facendo underfitting le nostre predizioni saranno meno influenzate dai dati in input e quindi avranno una bassa variance ed un alto Bias, poiché lo stesso errore si ripeterà sistematicamente.
Indicando nel cerchio rosso il valore corretto della y e con i pallini blu le nostre predizioni, possiamo graficare diverse combinazioni di Bias e Variance nel seguente modo

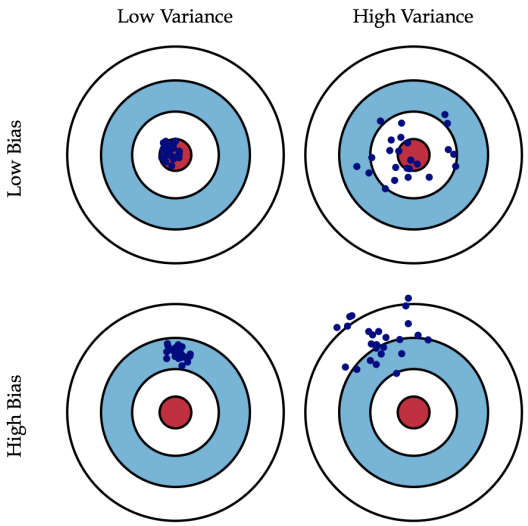

Figura 11: Interpretazione grafica Bias/Variance

9.1. Risultati ottenuti

Nel problema preso in considerazione si può osservare come i modelli più semplici (con meno parametri), ad esempio la Logistic Regression con

Boundary Lineare e Regolarizzazione, riescano ad ottenere risultati migliori in termini di Variance, proprio perchè meno complessi ed adatti a dataset con pochi campioni come nel nostro caso. Per quanto riguarda il bias, si nota che i valori maggiori si ottengono nella Linear Discriminant Analysis e nella Quadratic Discriminant Analysis, poichè molto probabilmente i campioni non sono distribuiti come delle normali.

Modello	Bias	Variance
Linear Discriminant Analysis	0.0341	0.0049
Quadratic Discriminant Analysis	0.0469	0.0026
Diagonal Discriminant Analysis	0.0469	0.0026
Logistic Regression (LB)	0.0164	0.0056
Logistic Regression (QB)	-0.0210	0.0286
Logistic Regression (LBR)	0.0287	0.0029
Logistic Regression (QBR)	-0.0232	0.0109

Tabella 2: errori minimi e medi dei modelli utilizzati

10. Conclusioni

Abbiamo notato che almeno in questo particolare esempio, non vi sia una gran differenza tra la scelta di un classificatore generativo, piuttosto che uno discriminativo. La differenza la fa la complessità dell'algoritmo scelto; per il dataset preso in questione, essendo povero di campioni, è stato molto meglio orientarsi su algoritmi che tendono a non utilizzare polinomi di grado troppo elevato e che quindi riescono a generalizzare meglio in questi casi.
Una giustificazione ai valori dell'error rate così elevati, va cercata nella confidenzialità dei dati. Cercando il motivo dell'alta percentuale di errore, abbiamo trovato una relazione da parte di due studenti dell'università di Standford, che utilizzando il nostro stesso dataset con algoritmi molto più complessi come ad esempio le reti neurali, le quali cercano di fittare un classificatore non lineare, sono arrivati ad avere error rate non minori dell' 8%.

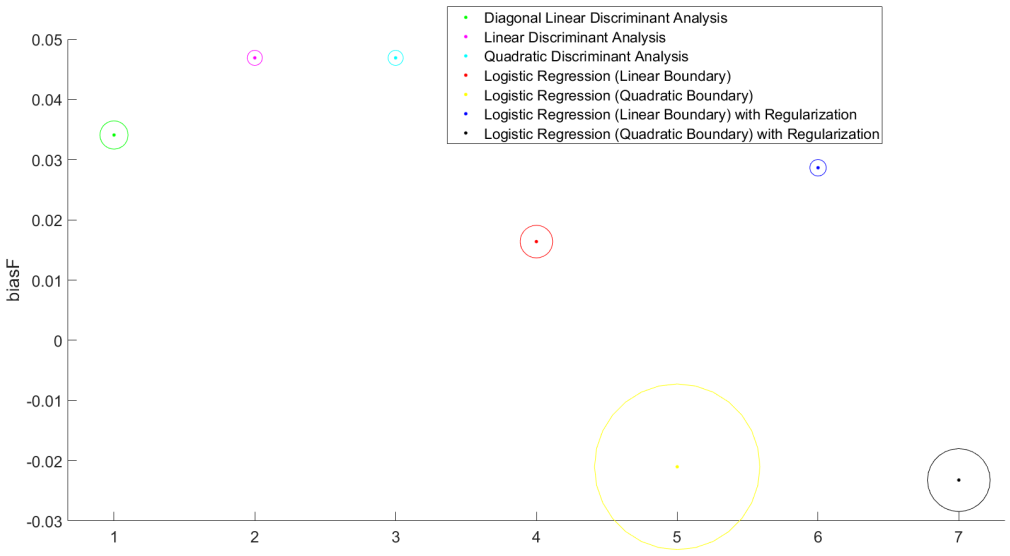

Figura 12: Bias vs Variance

Riferimenti bibliografici

[1] John Ross Quinlan
Credit Approval Data Set
http://archive.ics.uci.edu/ml/datasets/Credit+Approval

[2] Håkon Hapnes Strand
One Hot Encoding
https://www.quora.com/What-is-one-hot-encoding-and-when-is-it-used-in-data-science