

Indice

1	Analisi dei dati di probabilità	2
1.1	Problema in esame	2
1.2	Caratteristiche degli eventi di coppia	2
1.2.1	Eventi indipendenti	3
1.2.2	Eventi dipendenti	3
1.3	Evento conosciuto ed evento indovinato	4
1.3.1	Probabilità di rispondere correttamente ad una domanda	5
1.3.2	Il piano	7
2	Rete neurale	12
2.1	Test effettuati	13
2.1.1	Architettura della rete: 4 neuroni per ciascuno dei 2 layers	14
2.1.2	Architettura della rete a 2 neuroni per ciascuno dei 2 layers	15
2.1.3	Architettura della rete a 4 neuroni per 1 layer	18
2.2	Sviluppo della rete delle domande nel database aziendale	19
2.2.1	Montaggio e configurazione della rete	19
2.2.2	Test e Documentazione	22
3	Principal Component Analysis	40
3.1	Metodologia applicata	40
3.2	Sviluppo	41
3.3	Risultati ottenuti	43
3.3.1	Calcolo della Principal Component Analysis	43
3.3.2	Calcolo degli autovettori ed individuazione di Summary	44
3.3.3	Calcolo degli autovettori	52
3.3.4	Calcolo della matrice di correlazione	52
3.4	Conclusione dell'analisi	55

1 Analisi dei dati di probabilità

Durante il periodo 20/05 - 24/05 mi sono occupata di analizzare la probabilità che ha un candidato di rispondere correttamente alle domande in fase di test; valutando le relazioni di dipendenza che possono esistere tra più domande e l'impatto che può assumere la fortuna.

1.1 Problema in esame

Test, sottoposto ad un candidato durante un colloquio, composto da *domande a tripla risposta multipla*.

Nel suddetto documento vengono analizzate le relazioni che intercorrono tra due domande, denominate A e B, a seconda se il candidato risulta in grado di rispondervi correttamente o meno.

1.2 Caratteristiche degli eventi di coppia

Tipi di eventi trattati:

- **Eventi indipendenti;**
- **Eventi dipendenti:**
 - A e B sono strettamente dipendenti;
 - A implica B.
- **Evento conosciuto ed evento indovinato.**

Struttura usata per rappresentare la probabilità degli eventi di coppia:

AB

\wedge

$A\ B$

\vee

Z

con:

- AB rappresenta la probabilità complessiva dell'evento che si verifica sempre;
- A rappresenta la probabilità che permette il verificarsi di A, ma non di B;

- B rappresenta la probabilità che permette il verificarsi di B, ma non di A;
- Z rappresenta la probabilità a zero, l'impossibilità del verificarsi dell'evento.

1.2.1 Eventi indipendenti

A e B sono due domande la quali risposte sono completamente scorrelate tra di loro.

$$\begin{array}{c}
 P(A)P(B) \\
 /\backslash \\
 P(A)(1 - P(B)) \quad (1 - P(A))P(B) \\
 \backslash/ \\
 (1 - P(A))(1 - P(B))
 \end{array}$$

Considerazioni generali La probabilità complessiva nel caso di domande indipendenti A e B viene data da $P(A)$ per $P(B)$.

Se è conosciuta dal candidato la risposta alla domanda A ma non alla domanda B la probabilità di ottenere una risposta corretta è $P(A)$, mentre la probabilità di ottenere una risposta non corretta per B vale $1 - P(B)$. Il ragionamento duale è svolto nel calcolo della probabilità per la risposta corretta alla domanda B ma non ad A.

La probabilità di non ottenere alcuna risposta corretta alle due domande viene calcolata prendendo in considerazione gli eventi contrari a quelli coinvolti. Dunque per A la probabilità che il candidato non conosca la soluzione è $1 - P(B)$, dualmente per B la probabilità è $1 - P(A)$.

1.2.2 Eventi dipendenti

A e B sono due domande fortemente correlate tra di loro se si risponde correttamente ad una delle due domande si risponde correttamente anche all'altra.

$$\begin{array}{c}
 P(A)^2 \\
 /\backslash \\
 0 \quad 0 \\
 \backslash/ \\
 (1 - P(A))^2
 \end{array}$$

Considerazioni generali La probabilità complessiva nel caso di domande dipendenti A e B viene data da $P(A)$ per $P(B)$; ma $P(A) = P(B)$ dunque $P(A)^2 = P(B)^2$.

Conseguentemente se il candidato non conosce la risposta alla domanda A non può conoscere la risposta alla domanda B per cui la probabilità di conoscere uno dei due eventi è pari a 0.

In questo caso la probabilità a 0 è $(1 - P(A))(1 - P(B)) = (1 - P(A))^2$ essendo che $A=B$.

A implica B Se si sa rispondere alla domanda A di conseguenza si è in grado di rispondere anche alla domanda B.

Tuttavia non vale il ragionamento opposto, se si sa rispondere alla domanda B non significa che si è in grado di rispondere alla domanda A.

$$P(A)$$

$$\wedge$$

$$0 \quad P(B) - P(A)$$

$$\vee$$

$$1 - P(B)$$

Considerazioni generali La probabilità complessiva nel caso di domande dipendenti A e B viene data esclusivamente da $P(A)$ in quanto la conoscenza di sia di A che di B è possibile solo se si ha piena conoscenza di A.

Dunque la probabilità che si conosca la risposta alla domanda A ma non a B è impossibile (pari a 0); mentre se si ha conoscenza della domanda B ma non di A la probabilità si stanziava a $P(B) - P(A)$.

La probabilità a zero è $1 - P(B)$ indicatore dell'impossibilità di avere la risposta corretta per A.

1.3 Evento conosciuto ed evento indovinato

Durante un test il candidato deve saper scegliere la risposta, corretta o meno, alla domanda posta. Le variabili che entrano in gioco durante l'esecuzione dell'atto non riguardano esclusivamente la conoscenza personale del singolo. La probabilità di un evento A è data dalla formula:

$$P(A) = P(A_C) + P(A_I)$$

Le variabili in uso sono:

- $P(A_C)$: probabilità che il candidato sappia rispondere alla domanda A correttamente per sua conoscenza;
- $P(A_I)$: probabilità che il candidato sappia rispondere alla domanda A correttamente indovinando.

Per quanto appena definito sopra valgono le seguenti proprietà:

1. $P(B_C|A_C) = 1$
2. $P(B_C|A_I) = P(B_C)$
3. $P(B_I|A_C) = 0$
4. $P(B_I|A_I) = P(B_I)$

1.3.1 Probabilità di rispondere correttamente ad una domanda

Variabili coinvolte:

- $P(A)$: probabilità necessaria perchè si verifichi, per la domanda A, che il candidato dia la risposta corretta. Per la legge dei grandi numeri la frequenza porta alla probabilità.
- S_0 : insieme dei casi in cui in un domanda non viene scartata alcuna risposta dal dominio delle risposte possibili;
- S_1 : insieme dei casi in cui in una domanda viene scartata una risposta dal dominio delle risposte possibili;
- S_2 : insieme dei casi in cui in una domanda vengono scartate due risposte dal dominio delle risposte possibili.
- $P(I)$: probabilità di dare la risposta corretta alla domanda A indovinando;
- $P(C)$: probabilità di dare la risposta corretta alla domanda A per conoscenza.

Sapendo che $P(I) = P(A) - P(C)$ logicamente vale anche $P(A) = P(I) + P(C)$.

Se un candidato non è in grado scartare alcuna risposta dalla domanda ha 1 possibilità su 3 di, indovinando, dare la risposta corretta. Se un candidato invece risulta in grado di scartare una risposta, sbagliata, alla domanda rimane con 1 possibilità su 2 di poter dare la risposta corretta. Se invece, caso ottimo, il candidato ha piena conoscenza della domanda posta risulta in grado di scartare due risposte sbagliate lasciando un'unica risposta possibile, quella esatta. Il ragionamento sopra espresso può venire espresso con la seguente espressione:

$$P(A) = P(S_0)\frac{1}{3} + P(S_1)\frac{1}{2} + P(S_2)$$

Ora individuiamo quale è la probabilità effettiva per un candidato di dare la risposta corretta ad una domanda A.

$$\begin{aligned} 1 &= S_0 + S_1 + S_2 \\ S_0 &= 1 - S_1 - S_2 \end{aligned}$$

Sostituendo:

$$\begin{aligned} P(A) &= (1 - P(S_1) - P(S_2))\frac{1}{3} + P(S_1)\frac{1}{2} + P(S_2) \\ &= \frac{1}{3} - \frac{1}{3}P(S_1) - \frac{1}{3}P(S_2) + \frac{1}{2}P(S_1) + P(S_2) \\ &= \frac{1}{3} + \frac{1}{6}P(S_1) + \frac{2}{3}P(S_2) \end{aligned}$$

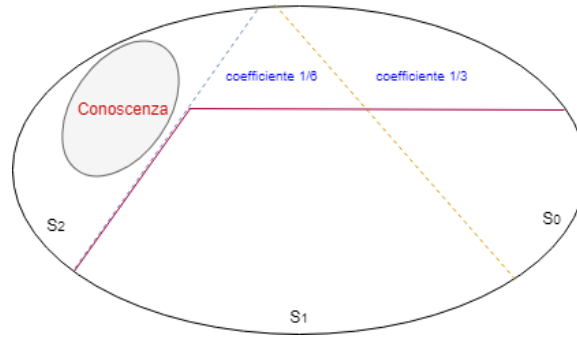


Figura 1: Rappresentazione insiemistica della probabilità di rispondere correttamente ad una domanda: $P(A)$

Considerazioni importanti

In conclusione $P(A) = \frac{1}{3} + \frac{1}{6}P(S_1) + \frac{2}{3}P(S_2)$. Ovvero la probabilità per un candidato di dare in una domanda A la risposta corretta dipende dai seguenti fattori:

- $\frac{1}{3}$: coefficiente che rappresenta la probabilità effettiva per chi non conosce la risposta alla domanda di dare la risposta corretta;
- $\frac{1}{2}P(S_1)$: coefficiente che rappresenta la probabilità effettiva di dare la risposta corretta quando il candidato è in grado di scartare una risposta sbagliata alla domanda;
- $\frac{2}{3}P(S_2)$: coefficiente che rappresenta la probabilità effettiva di dare la risposta corretta quando il candidato è in grado di scartare due risposte sbagliate alla domanda.

Dall'analisi della tipologia di eventi di coppia e dal calcolo della probabilità necessaria per poter rispondere correttamente ad una domanda, si è giunti alla valenza dei seguenti assiomi:

1. Le coppie di domande A e B devono essere fra loro disgiunte, altrimenti si genererebbero situazioni di invalidità dei risultati;
2. Per rispondere correttamente ad una domanda non è necessario che il candidato abbia piena conoscenza di tutti gli argomenti richiesti dall'esame, ma bensì ne risultano sufficienti $n - 1$;
3. La probabilità di conoscere è contenuta all'interno di S_2 , in quanto se un candidato conosce è conseguentemente in grado, da una domanda, di scartare due risposte sbagliate.

1.3.2 Il piano

La probabilità $P(A)$ che un candidato ha in gioco nel momento in cui si approccia a rispondere ad una domanda può venire rappresentata in un piano.

Di seguito viene mostrata l'immagine di un modellino, rappresentativo di $P(A)$, realizzato durante l'analisi.

TODO: foto modello

Ognuno dei tre assi cartesiani rappresenta un insieme dei casi di scarto (S_0 , S_1 , S_2). L'intersezione tra i punti del piano indica la regione accettabile contenente il range di valori assumibili da $P(A)$. Tale punto proiettato su ognuno dei tre assi permette l'individuazione esatta dei coefficienti delle variabili S_0 , S_1 , S_2 .

Ogni porzione del piano viene individuata con la seguente tecnica:

1. Per individuare ogni retta passante per S_0 , S_1 e S_2 è necessario assumere che $S_0 + S_1 + S_2 = 1$;
2. La retta passante per S_0 è rappresentabile per mezzo delle seguenti equazioni:

$$S_0 = 0 \text{ e } S_1 + S_2 = 1$$

In questo modo l'asse S_0 è fissato a 0 e estrapolando S_1 e S_2 da $S_1 = -S_2 + 1$ assumono valori tra (0,1).

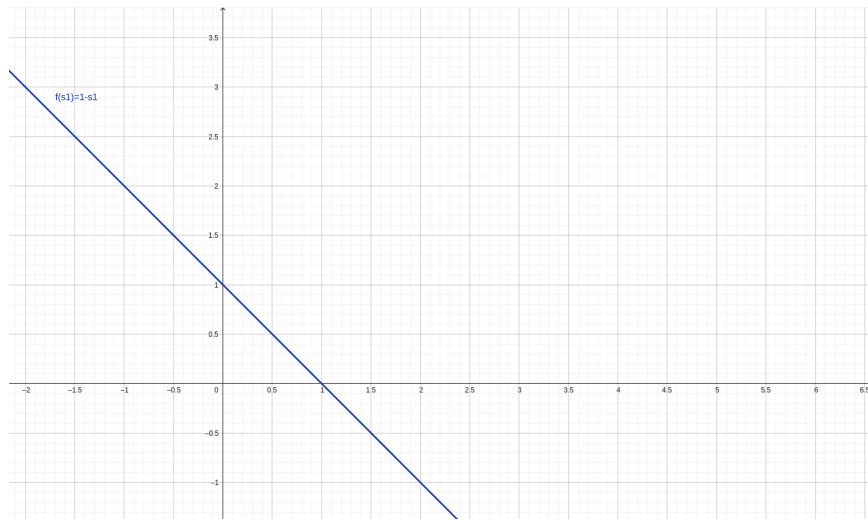


Figura 2: Rappresentazione della retta passante per $S_0 = 0$

3. Il medesimo ragionamento vale per le rette passanti per S_1 e S_2 .

$$S_1 = 0 \text{ e } S_0 + S_2 = 1$$

l'asse S_1 è fissato a 0 e S_0 e S_2 assumono valori tra (0,1).

$$S_2 = 0 \text{ e } S_1 + S_0 = 1$$

l'asse S_2 è fissato a 0 e S_0 e S_2 assumono valori tra (0,1).

4. In questo modo l'unione di tutte le rette passanti per gli assi creano la regione accettabile dei valori di $P(A)$.

Avendo rappresentato il piano si ottiene nei punti di intersezioni fra le tre rette la regione accettabile per $P(A)$. Inoltre è possibile, ora, individuare il fascio di rette che tangenti il piano permettono di affermare se una specifica domanda è, in base alla sua frequenza, ha difficoltà bassa, media, alta per un candidato.

- Se una domanda ha una difficoltà bassa la retta si situa passante per i punti $0 < S_2 \leq 1$ (molto vicino a 1) e $(S_0, S_1) < 0$ (tendenti a 0);
- Se una domanda ha una difficoltà alta la retta si situa passante per i punti $S_2 \leq 0$ (molto vicino a 0), $S_1 < 1$ e $S_0 \leq 1$ (tendente a non scartare alcuna risposta);

- Se una domanda ha una difficoltà media la retta si situa nella parte centrale della regione accettabile, passante per i punti $0 \leq (S_0, S_1, S_2) \leq 1$.

Rappresentazione di P(A)

Vediamo alcuni casi di come le domande possono venire rappresentate sul piano:

La funzione di partenza è:

$$F = \frac{1}{3} + \frac{1}{6}S_1 + \frac{2}{3}S_2$$

Va esplicitato S_1 , i passaggi utili da fare sono i seguenti:

$$\frac{-1}{6}S_1 = \frac{1}{3} + \frac{2}{3}S_2 - F \rightarrow S_1 = -4S_2 - 2 + 6F$$

Essendo che $0 \leq S_2 \leq 1$ usando $S_1 = 1$ e $S_2 = 0$ allora si ottiene che $F = \frac{1}{2} = 0.5$

Quanto appena calcolato può venire rappresentato graficamente impiegando la retta $S_1 = 1 - S_2$ (responsabile di definire una porzione del piano in base alle variabili coinvolte) e mediante la retta $S_1 = -4S_2 - 2 + 6F$ (che permette di calcolare il fascio di rette tangenti alla prima retta).

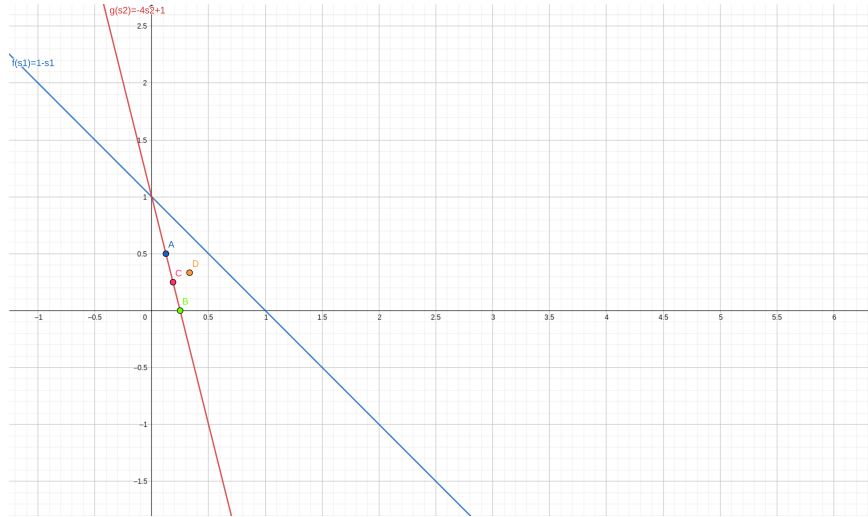


Figura 3: Rappresentazione di P(A) per una frequenza 0.5 proiettata su assi $S_0 = 0$, S_1 e S_2 .

Nella figura sopra sono rappresentati i seguenti significati:

- La linea azzurra rappresenta $S_2 = 1 - S_1$;
- La linea rosa rappresenta la retta tangente $S_1 = -4S_2 + 1$;
- Punto A (blu):

$$S_1 = 0.5 = \frac{1}{2}$$

$$\frac{1}{2} = -4S_2 + 1 \rightarrow 4S_2 = 1 - \frac{1}{2} \rightarrow S_2 = \frac{1}{8}$$

$$S_0 = 1 - \frac{1}{2} - \frac{1}{8} = \frac{3}{8}$$

Ovvero metà dei candidati sottoposti alla domanda sa scartare una delle risposte, lo 0.16% sa dare la risposta corretta e lo 0.36% non sa scartare alcune delle risposte possibili.

- Punto B (verde):

$$S_1 = 0$$

$$S_2 = \frac{1}{4}$$

$$S_0 = 1 - \frac{1}{4} = \frac{3}{4}$$

Ovvero nessun dei candidati sottoposti alla domanda sa scartare una delle risposte, lo 0.25% sa dare la risposta corretta e lo 0.75% non sa scartare alcune delle risposte possibili.

- Punto C (fucsia):

$$S_1 = \frac{1}{4}$$

$$S_2 = \frac{3}{16}$$

$$S_0 = 1 - \frac{1}{4} - \frac{3}{16} = \frac{9}{16}$$

Ovvero lo 0.25% dei candidati sottoposti alla domanda sa scartare una delle risposte, lo 0.19% sa dare la risposta corretta e lo 0.56% non sa scartare alcune delle risposte possibili.

- Punto D (arancione):

$$S_1 = \frac{1}{3}$$

$$S_2 = \frac{1}{3}$$

$$S_0 = 1 - \frac{1}{3} - \frac{1}{3} = \frac{1}{3}$$

Osserviamo che il punto in esame fuoriesce dalla regione delimitata dalla retta tangente di frequenza 0.5 ($S_1 = -4S_2 + 1$). Conseguenza diretta data dall'impossibilità di ottenere una probabilità del 50% sulla domanda con $\frac{1}{3}$ di candidati che sa scartare 2 risposte, $\frac{1}{3}$ che ne sa scartare 1 e $\frac{1}{3}$ nessuna.

Vediamo ulteriori due esempi che permettono di valutare cosa accade nel piano nel caso di una frequenza:

1. Quasi in prossimità di 1;

2. Pari alla soglia minima dell'indovinato.

Il grafico è il seguente:



Figura 4: Rappresentazione di $P(A)$ per una frequenza 0.33 e 0.8 proiettate su assi $S_0 = 0$, S_1 e S_2 .

- La linea azzurra mostra la retta tangente con frequenza 0.80%. In questa abbiamo calcolato il punto E (giallo):

$$\begin{aligned} S_1 &= 0 \\ S_2 &= \frac{14}{20} \\ S_0 &= 0 \end{aligned}$$

Quasi la totalità dei candidati ha la conoscenza per poter scartare tutte le risposte sbagliate e dare la risposta giusta alla domanda.

- La linea viola mostra la retta tangente con frequenza 0.33%. In questa abbiamo calcolato il punto F (rosa):

$$\begin{aligned} S_1 &= 0 \\ S_2 &= 0 \\ S_0 &= 1 \end{aligned}$$

Ovvero nessuno dei candidati ha la conoscenza per poter scartare nè una nè due risposte, per cui l'unica possibilità per un candidato di rispondere alla domanda è indovinare. È evidente come se un candidato non sa la risposta ad una domanda ha una probabilità dello 0.33% di poter indovinare la risposta corretta.

2 Rete neurale

La libreria utilizzata per sviluppare la Rete neurale è stata *ConvNetJS*. L'aspetto positivo di tale scelta è stata la semplicità nell'utilizzo del linguaggio javascript; l'aspetto negativo ha riguardato la totale mancanza di mantenibilità della libreria stessa che comporta la scarsità di esempi applicativi, oltre alla documentazione ufficiale, che costringono lo sviluppatore ad una ricerca approfondita personale in un ambiente ove le nozioni si presentano scarse e a continue prove per verificare la validità del codice prodotto.

Questions test - Prevision Neural Net

CONFIGURAZIONE RETE NEURALE

```
layer_defs = [];  
layer_defs.push({type:'input', out_sz:1, out_sy:1, out_depth:6});  
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});  
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});  
layer_defs.push({type:'regression', num_neurons:6});  
  
net = new convnetjs.Net();  
net.makeLayers(layer_defs);  
  
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01, momentum:0.1, batch_size:10, l2_decay:0.001});
```

OUTPUT DELLA RETE

```
Richiesta di previsione inoltrata alla rete ...  
Richiesta di previsione accettata  
Il vettore [0,1,0,0,0,0] ha previsione calcolata di  
[-0.14178025275470646,-0.014639237878831693,-0.00131295482446743,0.11790470015357472,0.14410273493990078,0.1329311955165462]  
Rete Neurale in attesa ... Inserire risultati del test...  
Inserimento risposte alle domande del test andato a buon fine  
Ricapitolazione dati inseriti: Risposte ottenute [0,-1,0,0,0,0]  
Inizio allenamento della rete ...  
Richiesta di previsione inoltrata alla rete ...  
Richiesta di previsione accettata  
Il vettore [0,1,1,0,0,0] ha previsione calcolata di  
[-0.9703841414997074,-0.7824502014273779,0.34748891066895563,-0.8448991676224333,-0.5825080715009475,0.29369850449980084]  
Rete Neurale in attesa ...
```

INSERIMENTO DATI DI ADDESTRAMENTO

Number of fields (max 9999): [Insert fields](#)

Input1 Input2 Input3 Input4 Input5 Input6

[save](#)

[start trainer](#)

Vettore di previsione

[prevision trainer](#)

Figura 5: Interfaccia utente della Rete neurale di prova.

Durante il periodo 24/05 - 31/05 mi sono occupata dello sviluppo di una Rete neurale in grado di ricevere in input un training set di dimensione 6 e di restituire una previsione sui dati di apprendimento ricevuti. Il problema che la rete mira ad analizzare è quello discusso nel precedente capitolo *Analisi dei dati di probabilità*

Per agevolare l'apprendimento della rete, ed ottenere delle previsioni stabili mi sono occupata di implementare due metodi di generazione randomica di dati in modo da far apprendere massicciamente la stessa. Il dato prodotto consiste in un vettore di 6 elementi, composto da -1, 0 e 1 con il seguente criterio:

- **-1**: la domanda x è stata posta al candidato che ha risposto in maniera errata;
- **0**: la domanda x non è stata posta al candidato;
- **1**: la domanda x è stata posta al candidato che ha saputo rispondere correttamente.

Il primo metodo che ho sviluppato si occupa di generare un vettore di dati di apprendimento basandosi esclusivamente su come le domande sono interconnesse tra di loro (grazie all'uso di un grafo della conoscenza costruito ad hoc); il secondo metodo ripropone quanto perseguito dal primo metodo con il valore aggiunto di generazione di un profilo randomico di un candidato, che tiene conto della probabilità di risposta ad una domande seguendo la formula $P(A) = \frac{1}{3} + \frac{1}{6}P(S_1) + \frac{2}{3}P(S_2)$.

2.1 Test effettuati

Alcune decisioni che ho preso durante la scelta dell'architettura della rete riguardano i seguenti settori:

1. Una rete neurale non deve, per fornire dei dati attendibili, possedere un numero di neuroni troppo elevato rispetto al trainset effettuato; altrimenti la previsione ritornerebbe l'identità del vettore di input della stessa, come conseguenza diretta della capacità troppo elevata di immagazzinare dati.
2. I layers, ho deciso, di allenarli mediante tecnica di regressione, che permette l'inserimento in input di una funzione obiettivo e l'ottenimento di un risultato, in output, anche in virgola mobile e composto di tanti elementi quanti sono i neuroni di regressione dichiarati. Per la mia rete di prova è necessario dichiarare 6 neuroni in regressione perchè l'output, appunto, che ci si aspetta dal sistema è di 6 elementi.
3. Per costruire un dataset di dati consistente che permettesse alla rete di imparare qualcosa ho costruito un grafo della conoscenza con lo scopo di mettere in relazione degli argomenti che coinvolgono uno o più domande.

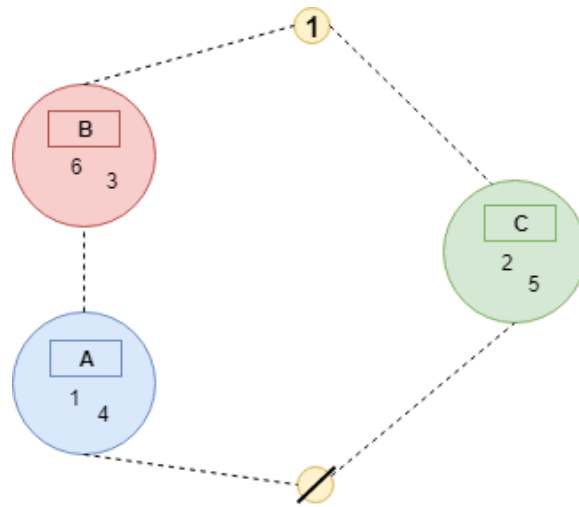


Figura 6: Grafo rappresentante le relazioni esistenti tra il set di domande di prova.

Per svolgere l'apprendimento ogni vettore, facente parte del dataset, viene dato in pasto alla rete che a sua volta provvede alla sua assimilazione come conoscenza mediante la tecnica dell'autoencoder, ovvero la rete impara il vettore riducendone lo spazio occupato.

4. Per creare il dataset ho ritenuto sufficiente generare *2000* vettori di risposta in modo da compiere in maniera esaustivo l'apprendimento della rete.

Il vettore passato in input per svolgere le previsioni è $[0,0,0,0,0,0]$, $[0,0,1,0,1,0]$ e $[0,0,-1,0,0,0]$

Le aspettative riguardano la previsione di risposta di un candidato . Di seguito riporto quanto è stato rilevato in fase di test.

2.1.1 Architettura della rete: 4 neuroni per ciascuno dei 2 layers

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:6});

net = new convnetjs.Net();
net.makeLayers(layer_defs);
```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

I layers utilizzati sono 2 e composti da 4 neuroni.

Training set standard a 4 neuroni per ciascuno dei 2 layers

- Il vettore [0,0,0,0,0,0] ha previsione calcolata di
[-0.021598804903572744,-0.1372509042342871,0.06611969158456255,
0.018121335417653706,-0.11264571886853292,0.17520370837747462]

Appaiono in relazione le domande 1, 2, 5 e 3, 4, 6.

Gli scostamenti tra le coppie 2, 5 e 3, 6 sono consistenti con quelle che sono le relazioni di dipendenza, invece 1, 4 ha una differenza di 0.016 circa che parte da qualche millesimo fino 0.5 Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4; nel test in analisi questo non viene rispettato da nessuna delle coppie in analisi per differenze che vanno da qualche millesimo fino a 0.018 circa.

Osservazioni

L'architettura testata si compone di 4 neuroni a layer su una base di 2000 test correndo il rischio di avere una rete che apprende troppo e come effetto negativo "veda" addirittura cose che non esistono. A prova di ciò sono i risultati non conformi alle attese. Dunque mi fermo qui con il test di tale architettura e riducendone il numero di neuroni presenti in ciascun layers e/o il numero di layers presenti.

Le nuove architetture su cui ho effettuato i test sono esposte nei paragrafi seguenti.

2.1.2 Architettura della rete a 2 neuroni per ciascuno dei 2 layers

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
```

```

layer_defs.push({type:'regression', num_neurons:6});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

```

I layers utilizzati sono 2 composti da 2 neuroni.

Training set standard su rete a 2 neuroni per ciascuno dei 2 layers

- Il vettore [0,0,0,0,0,0] ha previsione calcolata di
[0.31232372051574936,0.7253754889487585,-0.5051208979797573,
0.32075742158673093,0.7324947496336937,-0.4348299972940168]
Appaiono in relazione le domande 1, 2, 4, 5 e 3, 6.
Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti
con quelle che sono le relazioni di dipendenza fra le domande.
Le domande 3 e 6 si dovrebbero presentare con una positività
inferiore rispetto a 1 e 4; in questo test la regola viene rispettata
pienamente.
Dai dati della previsione si nota come il candidato ha una buona
probabilità di saper rispondere alla coppia 1 e 4, e ancora più
elevata di saper rispondere correttamente alla coppia 2 e 5; mol-
to bassa di saper rispondere correttamente alle 3 e 6 che sono,
appunto, di una difficoltà maggiore rispetto alla coppia 1 e 4.
- Il vettore [0,0,1,0,1,0] ha previsione calcolata di
[0.5123144717131076,0.9123354449531641,0.2837937822420923,
0.46449868699771607,0.9029832167165894,0.3227303792035435]
Appaiono in relazione le domande 1, 2, 3 4, 5, 6.
Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti
con quelle che sono le relazioni di dipendenza fra le domande.
Le domande 3 e 6 si dovrebbero presentare con una positività
inferiore rispetto a 1 e 4; in questo test la regola viene rispettata
pienamente.
Dai dati della previsione si nota come il candidato ha un'ottima
probabilità di saper rispondere alla coppia 2 e 5 (come imposto
dal vettore previsione), buona di saper rispondere alla coppie 3
e 6 (come imposto dal vettore previsione) e più che buona di
saper rispondere alle 1 e 4, che sono di una semplicità più elevata
rispetto alla 3 e 4.

- Il vettore $[0,0,-1,0,0,0]$ ha previsione calcolata di $[0.3698539826215957, 0.288907514487717, -0.8504159455662308, 0.3663192502433841, 0.2937448801761998, -0.7845589473185985]$
 Appaiono in relazione le domande 1, 2, 4, 5 e 3, 6.
 Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4; in questo test la regola viene rispettata pienamente.
 Dai dati della previsione si nota come il candidato ha una discreta probabilità di saper rispondere alla coppia 2 e 5, un pò meglio di saper rispondere alla coppie 1 e 4 e più di non saper saper rispondere alle 3 e 6 (come imposto dal vettore previsione).

Training set con generazione del profilo di un candidato e calcolo delle probabilità di risposta a 2 neuroni per ciascuno dei 2 layers

- Il vettore $[0,0,0,0,0,0]$ ha previsione calcolata di $[0.057781303506280995, 0.0513731100126314, -0.06600467867066256, 0.029940883111932555, -0.019564515397168573, -0.09570617900597932]$
 Appaiono in relazione le domande 1, 2, 4 e 3, 5, 6.
 Gli scostamenti tra la coppia 1, 4 e 3, 6 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande; invece per la coppia 2 e 5 i segni sono opposti con una differenza di 0.024. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4, la regola viene rispettata pienamente. Le anomalie riscontrate sono da ricondurre alla natura del vettore di training che si basa sul calcolo della probabilità di una risposta che sul grafo della conoscenza.
- Il vettore $[0,0,1,0,1,0]$ ha previsione calcolata di $[0.19494624113789977, 0.1712744021266377, 0.577963304906936, 0.781098215373483, 0.3774535909060714, 0.03617314870307162]$
 Appaiono in relazione le domande 1, 2, 3, 4, 5, 6.
 Gli scostamenti tra le coppie 1 e 4, 2 e 5, 3 e 6 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4, la regola non viene rispettata dalla domanda 1 in rapporto con la domanda per una differenza di 0.37 circa. Le anomalie riscontrate sono da ricondurre alla natura del vettore di training che si basa sul calcolo della probabilità di una risposta che sul grafo della conoscenza.
- Il vettore $[0,0,-1,0,0,0]$ ha previsione calcolata di $[0.09845785763965222, 0.015421380649956663, -0.5138068038427066, -0.4853190165287735, -0.22629262719814794, 0.0008152164571250502]$

Appaiono in relazione le domande 1, 2, 6 e 3, 4, 5.

Gli scostamenti tra le coppie 1, 4 e 2, 5 e 3, 6 per una differenza tuttavia trascurabile che oscilla dallo 0.2 allo 0.5. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4, la regola non vale per la coppia 6 e 4. Le anomalie riscontrate sono da ricondurre alla natura del vettore di training che si basa sul calcolo della probabilità di una risposta che sul grafo della conoscenza.

Osservazioni

Confrontando i risultati ottenuti dalla rete con i layers impostati a 4 neuroni con quanto emerso dai dati risultanti dalla rete a 2 neuroni emerge come l'architettura a 2 neuroni a layers è sicuramente quella che da i risultati attesi.

Quanto emerso di discordate dal secondo training set è come da aspettative da associare alla natura stessa della creazione del set di dati.

2.1.3 Architettura della rete a 4 neuroni per 1 layer

Architettura della rete utilizzata:

```
layer_defs = [];  
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});  
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});  
layer_defs.push({type:'regression', num_neurons:6});
```

```
net = new convnetjs.Net();  
net.makeLayers(layer_defs);
```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,  
momentum:0.1, batch_size:10, l2_decay:0.001});
```

Viene utilizzato un unico layer da 4 neuroni.

Training set standard su rete a 4 neuroni per 1 layer

- Il vettore [0,0,0,0,0,0] ha previsione calcolata di
[0.12202628618565468,0.08221724740100582,0.02233631914718809,
0.09586625658118901,0.05558075220027264,0.13443779128784109]

Appaiono in relazione le domande 1, 2, 3, 4, 5, 6.

Gli scostamenti tra le coppie 1, 4 e 3, 6 e 2, 5 sono consistenti con quelle che sono le relazioni di dipendenza fra le domande. Le domande 3 e 6 si dovrebbero presentare con una positività inferiore rispetto a 1 e 4;

in questo test la regola non viene rispettata dalla domanda 6.
Dai dati della previsione si nota come il candidato non ha una buona probabilità di saper rispondere alle domande e la domanda 6 non si presenta conforme alle aspettative.

Osservazioni

Rispetto a quanto osservato nei casi precedenti, ancora l'architettura che rispetta le attese è quella con 2 neuroni per 2 layers.

Tale conclusione ha perfettamente senso in quanto il grafo della conoscenza che ho usato come base per costruire i vettori di apprendimento è composto da 3 nodi (A, B, C) indicanti 3 neuroni; il quarto può venire valutato come un nodo della rete utile per parametri in entrata e in uscita.

Per estendere maggiormente la mia conoscenza della rete, ho provveduto ad aumentare progressivamente il numero di neuroni a layers e osservarne le interazioni. Svolgendo ciò mi sono accorta che il risultato ottenuto dalla previsione era il più possibile vicino al vettore previsione; conseguenza diretta di un numero eccessivo di neuroni dati alla rete per l'apprendimento rispetto al training set svolto, generatrice di una situazione di overfitting e non attendibilità dei dati raccolti. L'architettura a 1 e 2 neuroni invece presenta una buona capacità di previsione in quasi tutti i casi, però tende ad andare in overfitting, come riporto di seguito:

Il vettore $[0,0,0,0,0,1]$ ha previsione calcolata di
 $[0.5613347853884025, 0.8310670629630683, -1.03049430206139,$
 $0.5492731069379962, 0.5679700877862532, -0.8637707232817535]$

Il numero di neuroni non è sufficiente per memorizzare che la domanda 6 deve essere positiva, e comporta a cascata la correttezza anche delle domande 3, 4 e 1. La situazione si presenta simile se il layer con 1 neurone è posto al di sotto.

2.2 Sviluppo della rete delle domande nel database aziendale

2.2.1 Montaggio e configurazione della rete

Durante la settimana dal 03/06 al 07/06 la mia attività principale è stata il montaggio e la configurazione della Rete neurale inerente il database aziendale con dataset i colloqui ai candidati. Inoltre ho rivolto parte delle ore a modificare e ottimizzare quanto già implementato nella Rete di prova, in modo che ogni cosa implementata sulla rete del database è presenta anche in versione ridotta.

Per rendere più comprensibile le previsioni di probabilità ottenute, a seguito dell'addestramento della rete e della data in pasto del vettore previsione, ho realizzato un'immagine canvas in cui ogni domanda viene raffigurata con un quadrattino colorato in base alla previsione risultante (verde se a 1, bianco a 0, rosso a -1, gradazioni di bianco - verde e bianco - rosso per i valori intermedi).

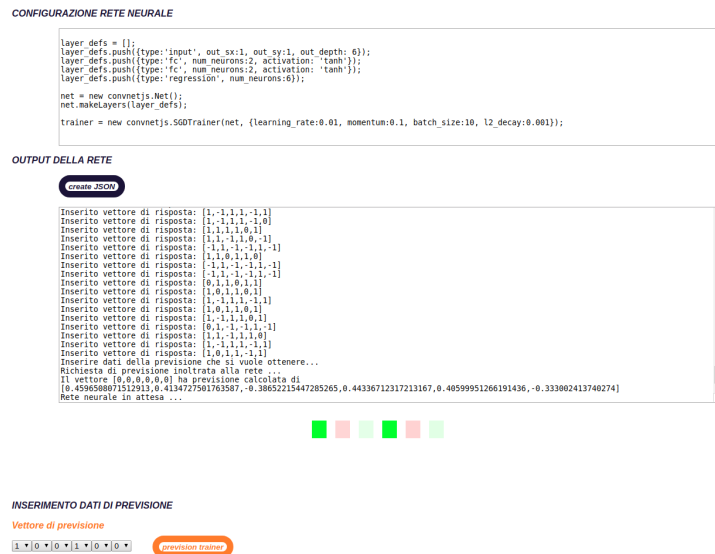


Figura 7: Rete di prova dopo lo sviluppo del canvas per le previsioni.

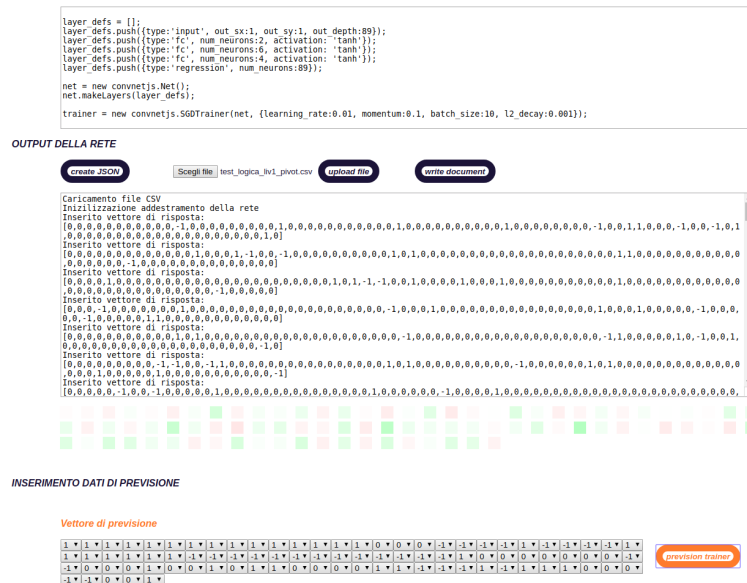


Figura 8: Rete neurale del database aziendale.

L'architettura che ho utilizzato, basandomi anche sul quanto appreso dalla rete neurale di prova e dal numero di vettori di test utilizzati (1245 vettori x 89) ho aumentato le dimensioni della rete.

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});
```

```
net = new convnetjs.Net();
net.makeLayers(layer_defs);
```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

Ho aggiunto un layer e messo un numero di neuroni per layer in modo da formare un romboide. Devo verificare la bontà di questa mia scelta o se invece mi porta ad una situazione di overfitting.

Analizzando il training set dei vettori ho riscontrato tali correlazioni:

- Solo una piccola parte delle domande presenti in un database vengono svolte durante un colloquio con un candidato, in media una decina su 89 possibili;

- Dalla rete sembra che le domande abbiano qualche correlazione, tuttavia la configurazione attuale ne rende difficoltosa l'individuazione.

2.2.2 Test e Documentazione

Durante la settimana dal 10/06 al 18/06 ho effettuato quanto definito al interno del Piano di Lavoro come "Test e Documentazione".

Test nella Rete di prova

Architettura della rete utilizzata:

```
layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:6});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:6});
```

```
net = new convnetjs.Net();
net.makeLayers(layer_defs);
```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});
```

- Il vettore [1,1,1,1,1,1] ha previsione calcolata di
[0.8521066399598267,0.898137375081856,0.9993098151218291,0.792190337086403,
0.811145866789799,0.9514731560722426]

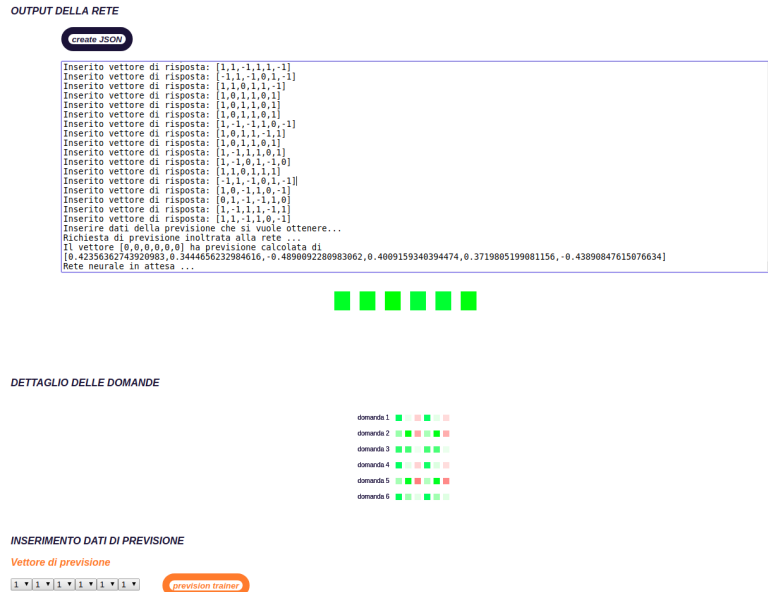


Figura 9: Risultato della rete di prova a seguito di un vettore di previsione $[1, 1, 1, 1, 1, 1]$ in input.

Quanto mostrato dal dettaglio delle domande ha il seguente significato per un candidato:

- se la domanda 1 è settata a 1 (corretta): la rete prevede che la domanda 1 e 4 abbiano una probabilità alta di essere risposte in modo corretto (verde); la 3 e 6 una probabilità non eccessiva di venire risposte in modo sbagliato (rosa attenuato), la 2 e la 5 di non venire nemmeno poste (bianco con qualche minima sfumatura di verde).
- se la domanda 2 è settata a 1 (corretta): la rete prevede che la domanda 1 e 4 abbiano una probabilità non molto alta di essere risposte in modo corretto (bianco con qualche sfumatura di verde); la 3 e 6 una buona probabilità di venire risposte in modo sbagliato (rosa), la 2 e la 5 di venire date in modo corretto (verde).
- se la domanda 3 è settata a 1 (corretta): la rete prevede che la domanda 3 e 6 abbiano una probabilità comunque bassa di essere risposte in modo corretto (bianco con qualche sfumatura di verde); la 1 e 4 con probabilità di venire risposte in modo corretto (verde) perchè più semplici delle domande 3 e 6, la 2 e la 5 di venire risposte correttamente (verde).
- se la domanda 4 è settata a 1 (corretta): la rete prevede un risultato identico a quanto ottenuto dalla domanda 1.

- se la domanda 5 è settata a 1 (corretta): la rete prevede un risultato simile a quanto ottenuto dalla domanda 2. Cambia solo quanto previsto dalle domande 3 e 6 che si presentano con un rosa un pò più intenso, in quanto con correlate alla coppia di domande 2 e 5.
 - se la domanda 6 è settata a 1 (corretta): la rete prevede un risultato simile a quanto ottenuto dalla domanda 3. La coppia 2 e 5 hanno una probabilità minore di essere date correttamente (bianco con sfumature di verde) ma perchè non correlate alle domande 3 e 6.
- Il vettore $[-1, -1, -1, -1, -1, -1]$ ha previsione calcolata di $[0.3440856175367477, -0.5026946644729329, -1.284368009920025, 0.35883842020377565, -0.37844446052773495, -1.1717763012412878]$

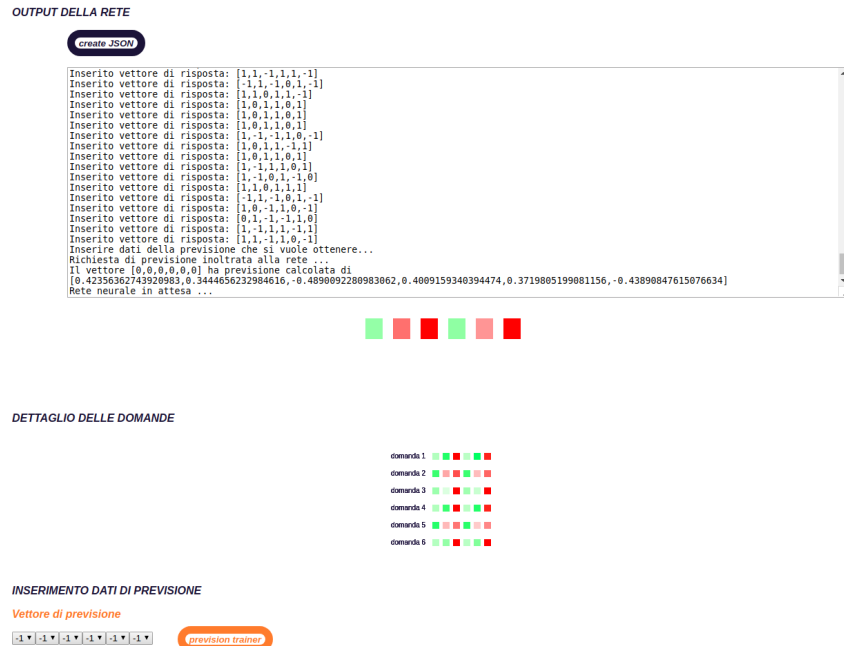


Figura 10: Risultato della rete di prova a seguito di un vettore di previsione $[-1, -1, -1, -1, -1, -1]$ in input.

- se la domanda 1 è settata a -1 (non corretta): la rete prevede che la domanda 1 e 4 non abbiano una probabilità alta di essere risposte in modo corretto (bianco con sfumature di verde); la 3 e 6 una probabilità molto alta di venire risposte in modo sbagliato (rosso), la 2 e la 5 di non venire nemmeno poste (verde con qualche sfumatura di bianco).

- *se la domanda 2 è settata a -1 (non corretta)*: la rete prevede che la domanda 1 e 4 abbiano una probabilità non molto alta di essere risposte in modo non corretto (verde con qualche sfumatura di bianco); la 3 e 6 una buona probabilità di venire risposte in modo sbagliato (rosa), la 2 e la 5 di venire date in modo non corretto (rosa molto attenuato).
- *se la domanda 3 è settata a -1 (non corretta)*: la rete prevede che la domanda 3 e 6 abbiano una probabilità comunque alta di essere risposte in modo non corretto (rosso); la 1 e 4 con bassa probabilità di venire risposte in modo corretto (bianco con qualche sfumatura di verde) perchè più semplici delle domande 3 e 6, la 2 e la 5 di non venire nemmeno poste o comunque basso di venire risposto correttamente (bianco con sfumature di verde).
- *se la domanda 4 è settata a -1 (non corretta)*: la rete prevede un risultato identico a quanto ottenuto dalla domanda 1.
- *se la domanda 5 è settata a -1 (non corretta)*: la rete prevede un risultato simile a quanto ottenuto dalla domanda 2. Cambia solo quanto previsto dalle domande 3 e 6 che si presentano con un rosa un pò meno intenso, in quanto non correlate alla coppia di domande 2 e 5.
- *se la domanda 6 è settata a -1 (non corretta)*: la rete prevede un risultato simile a quanto ottenuto dalla domanda 3. La coppia 2 e 5 hanno una probabilità maggiore di essere date correttamente (bianco con sfumature di verde) ma perchè non correlate alle domande 3 e 6.

Test nella Rete del database

Architetture testate

Durante tutto il periodo ho effettuato una serie di test su molteplici architettura della rete, con gradi di correlazione tra le domande pari al 100% o con una differenza massima di 5 punti colore rispetto al canvas risultante per ogni domanda.

Tuttavia non sono riuscita ad individuare un'architettura sufficientemente stabile per prevedere risultati attendibili; a causa della molteplicità di dati che hanno aumentato esponenzialmente la complessità di analisi rispetto alla Rete di prova, che non è calata neppure a seguito dell'analisi da parte mia delle domande presenti all'interno del database aziendale. Architettura della rete utilizzata:

- `layer_defs = []`;

```

layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

```

```

net = new convnetjs.Net();
net.makeLayers(layer_defs);

```

```

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

```

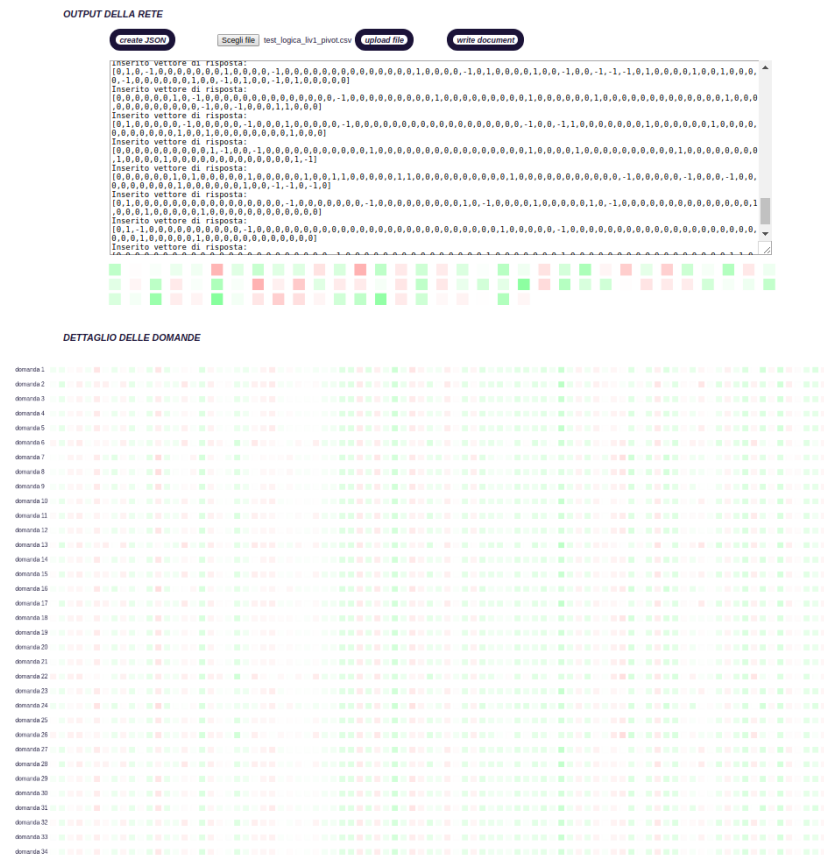


Figura 11: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

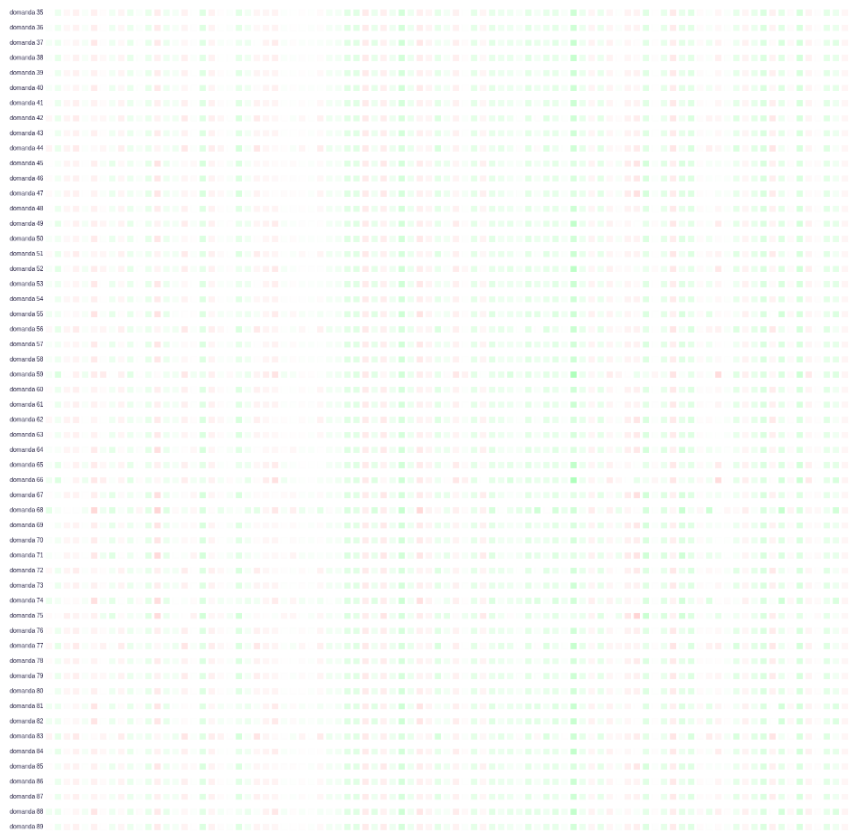


Figura 12: Risultato della rete del database a seguito di un vettore di previsione $[1, 1, 1, 1, 1, 1]$ in input.

Dagli screen della rete riportati sopra appare come "sembrano" domande:

1. Analisi verticale:

- *semplici* la 18, 22, 34, 35, 37, 39, 41, 44, 48, 50, 51, 52, 53, 54, 55, 56, 57, 59, 60, 67, 69, 71, 72, 77, 79, 80, 82, 84, 87 e 88. Inoltre di queste sembrano in relazione ancora più stretta le domande 18, 22, 40 e 59.
- *difficili* la 3, 4, 6, 13, 16, 19, 24, 25, 26, 36, 38, 42, 43, 46, 49, 61, 63, 65, 66, 70, 78, 81, 85 e 89. Inoltre di queste sembrano in relazione ancora più stretta le domande 6, 13, 19, 36, 38, 42, 46 e 70.

2. Analisi orizzontale: Appaiono in relazione stretta le seguenti domande:

- 2, 3, 4, 5;
- 7, 8, 9;

- 14, 16;
- 20, 21;
- 26, 32;
- 29, 32
- 33, 34, 35, 36, 38;
- 39, 41, 43;
- 46, 48;
- 49, 52;
- 50, 53;
- 72, 79;
- 81, 82;
- 86, 87, 88.



Figura 13: Risultato della rete del database a seguito di un vettore di previsione $[-1, -1, -1, -1, -1, -1]$ in input.

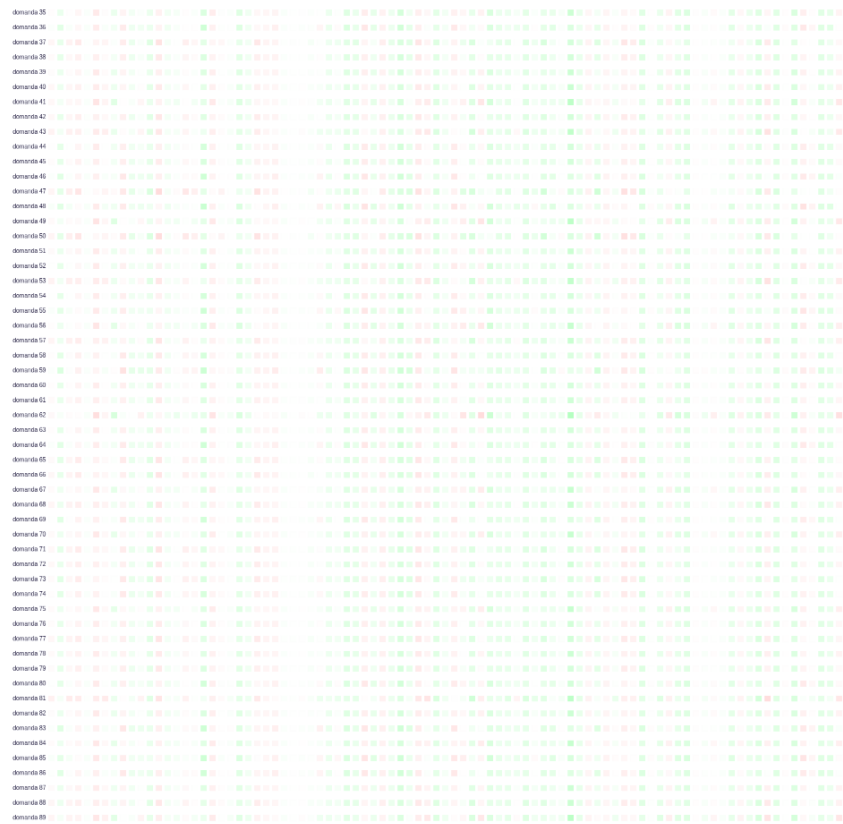


Figura 14: Risultato della rete del database a seguito di un vettore di previsione $[-1, -1, -1, -1, -1, -1]$ in input.

Dagli screen della rete riportati sopra appare come "sembrano" domande:

1. Analisi verticale:

- *semplici* la 18, 22, 34, 35, 37, 38, 39, 41, 44, 48, 50, 51, 52, 54, 55, 56, 67, 69, 71, 72, 77, 79, 80, 82, 84, 87 e 88. Inoltre di queste sembrano in relazione ancora più stretta le domande 18, 22, 38, 52 e 57.
- *difficili* la 3, 4, 6, 13, 19, 27, 36, 38, 42, 43, 46, 61, 62, 65, 70. Inoltre di queste sembrano in relazione ancora più stretta le domande 6, 13, 19, 36, 38, 42, 46, 61 e 62.

2. Analisi orizzontale: Appaiono in relazione stretta le seguenti domande:

- rimaste consistenti con il vettore $[1, 1, 1, 1, 1, 1]$.

• `layer_defs = []`;

```
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:12, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});
```

```
net = new convnetjs.Net();
net.makeLayers(layer_defs);
```

```
trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01, momentum:0.1, batch_size:100});
```

La prima cosa che ho notato è che aumentando il numero di neuroni sull'unico layer esistente, il valore della domanda corrispondente al vettore della previsione sembra sempre più marcato, segno che la rete "impara troppo" e ricade nel restituire l'immagine stessa del vettore previsione.



Figura 15: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

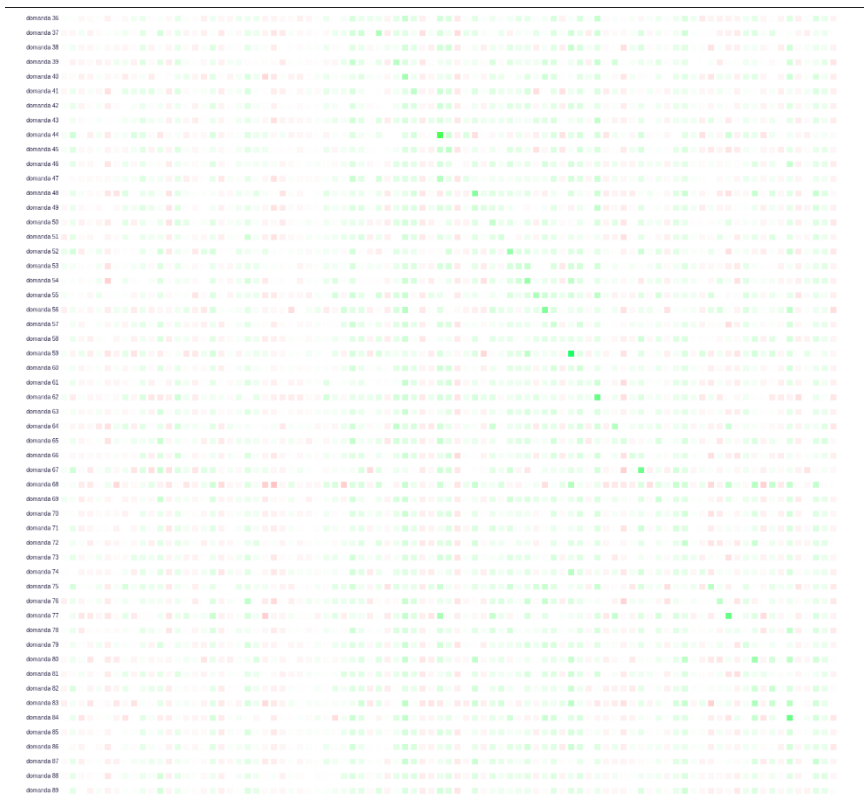


Figura 16: Risultato della rete del database a seguito di un vettore di previsione [1, 1, 1, 1, 1, 1] in input.

Dagli screen della rete riportati sopra appare come la situazione appare meno lineare del caso analizzato precedentemente. Le domande non vengono separate per linee rete; ma per aree di relazione.

Dagli screen della rete riportati sopra appare come "sembrano" domande:

1. Analisi verticale:

- *semplici* la 18, 22, 34, 35, 37, 39, 41, 44, 48, 50, 51, 52, 53, 54, 55, 56, 57, 59, 60, 67, 69, 71, 72, 77, 79, 80, 82, 84, 87 e 88. Inoltre di queste sembrano in relazione ancora più stretta le domande 18, 22, 40 e 59.
- *difficili* la 3, 4, 6, 13, 16, 19, 24, 25, 26, 36, 38, 42, 43, 46, 49, 61, 63, 65, 66, 70, 78, 81, 85 e 89. Inoltre di queste sembrano in relazione ancora più stretta le domande 6, 13, 19, 36, 38, 42, 46 e 70.

2. Analisi orizzontale: Appaiono in relazione stretta le seguenti domande:

– Vengono meno le relazioni individuate precedentemente.

```
• layer_defs = [];  
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});  
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});  
  layer_defs.push({type:'regression', num_neurons:89});  
  
  net = new convnetjs.Net();  
  net.makeLayers(layer_defs);  
  
  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,  
    momentum:0.1, batch_size:10, l2_decay:0.001});  
  
• layer_defs = [];  
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});  
  layer_defs.push({type:'fc', num_neurons:12, activation: 'tanh'});  
  layer_defs.push({type:'regression', num_neurons:89});  
  
  net = new convnetjs.Net();  
  net.makeLayers(layer_defs);  
  
  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,  
    momentum:0.1, batch_size:10, l2_decay:0.001});  
  
• layer_defs = [];  
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});  
  layer_defs.push({type:'fc', num_neurons:12, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:8, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});  
  layer_defs.push({type:'regression', num_neurons:89});  
  
  net = new convnetjs.Net();  
  net.makeLayers(layer_defs);  
  
  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,  
    momentum:0.1, batch_size:10, l2_decay:0.001});  
  
• layer_defs = [];  
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});  
  layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});  
  layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
```



```

layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001})

• layer_defs = [];
layer_defs.push
({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();

```

```

net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});

```

```

layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,

```

```

momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:5, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});

```

```

layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:18, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

```

```

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
    momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
    momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:3, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
    momentum:0.1, batch_size:10, l2_decay:0.001});

• layer_defs = [];
layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
layer_defs.push({type:'fc', num_neurons:2, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:6, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:4, activation: 'tanh'});
layer_defs.push({type:'fc', num_neurons:1, activation: 'tanh'});
layer_defs.push({type:'regression', num_neurons:89});

net = new convnetjs.Net();
net.makeLayers(layer_defs);

trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
    momentum:0.1, batch_size:10, l2_decay:0.001});

```

```

• layer_defs = [];
  layer_defs.push({type:'input', out_sx:1, out_sy:1, out_depth:89});
  layer_defs.push({type:'fc', num_neurons:8, activation: 'tanh'});
  layer_defs.push({type:'fc', num_neurons:8, activation: 'tanh'});
  layer_defs.push({type:'regression', num_neurons:89});

  net = new convnetjs.Net();
  net.makeLayers(layer_defs);

  trainer = new convnetjs.SGDTrainer(net, {learning_rate:0.01,
momentum:0.1, batch_size:10, l2_decay:0.001});

```

3 Principal Component Analysis

La **Principal Component Analysis** (acronimo PCA) è una tecnica impiegata nell'ambito della statistica multivariata¹ usata per semplificare i dati d'origine.

Lo scopo che tale tecnica persegue è lo studio della relazione esistenti tra i campioni d'interesse, con la riduzione di un numero più o meno elevato di variabili. La riduzione dimensionale avviene tramite una trasformazione lineare delle variabili coinvolte con lo scopo di effettuare la proiezione di quelle originarie in un nuovo sistema cartesiano, in cui tutte le variabili vengono ordinate in maniera decrescente per ordine di varianza. Successivamente la variabile con maggiore varianza viene proiettata sul primo asse, la seconda sul secondo e via via sempre così per tutte le variabili sotto esame. La PCA è particolarmente utile quando la dimensionalità dello spazio delle misure è elevata (molte colonne); tuttavia i campioni si vengono a trovare in uno spazio di dimensioni significativamente ridotte.

Indispensabile per la buona riuscita della tecnica è, come già esposto sopra, la ricerca del numero di componenti principali significative, ovvero tutte le variabili coinvolte a meno di quelle legate al rumore, che concorrono a comporre la *dimensionalità intrinseca*. Il "rumore" è sempre concentrato nelle ultime variabili, non includerle nell'analisi dei dati porta a dati più puliti, con un rapporto segnale/rumore più alto.

Il calcolo di quali sono le componenti più significative si ottiene con la varianza. La prima componente principale spiega la massima percentuale della variabilità presente nei dati rappresentabili in una singola dimensione, in poche parole la direzione lungo cui si registra la massima dispersione dei dati (quanto il valore medio si discosta). La varianza porta il vantaggio di essere indipendente dal sistema di riferimento, difatti conseguentemente una rotazione degli assi mantiene immutata la varianza totale all'interno del sistema.

3.1 Metodologia applicata

L'analisi e l'implementazione della tecnica di **Principal Component Analysis** l'ho svolta nel periodo 17 - 21 luglio con la seguente metodologia:

- Studio e configurazione di R e R-Studio;
- Studio del significato della tecnica e sua correlazione con le Rete neurali;
- Studio della metodologia di implementazione;
- Scelta del package più adeguato per svolgere l'applicazione della PCA sui dati di trainset;

¹parte della statistica in cui l'oggetto dell'analisi è almeno composta da due elementi

- Implementazione del metodo della PCA e dei metodi di supporto ad analisi dei dati.
- Analisi dei risultati ottenuti dalla PCA;
- Analisi delle domande presenti nel database aziendale;
- Confronto dei risultati della PCA con quelli ottenuti dalla Rete neurale e confronto della loro validità con le domande del database.

Ogni implementazione e osservazione l'ho svolta prima nei dati del trainset di prova e solo successivamente gli traslati all'interno del trainset delle domande nel database aziendale. In modo ho avuto sempre ben la correttezza e le possibili correlazioni vigenti nel metodo in uso.

3.2 Sviluppo

La tecnica di PCA effettua analisi dei dati. I dati che ho impiegato sono i medesimi che ho utilizzato per effettuare il trainset della Rete neurale. Il formato utilizzato è sempre CSV e i dati contenuti sono strutturati per riga con i risultati di ogni test e per colonna con il risultato di una specifica domanda k.

1. Caricamento del package **factoextra**: la mia scelta è ricaduta su tale package e non in altri con la medesima funzione, per via della sua visualizzazione dei dati elegante basata su ggplot2;
2. **Caricamento del file CSV** in memoria per mezzo della trasposizione in `ata frame`;
3. *Standardizzazione* dei dati del trainset. Tale compito mi è risultato indispensabile perchè anche se di norma, la standardizzazione, viene usata per evitare situazioni erronee (alcune variabili X presentate con una variabilità molto maggiore rispetto ad altre) e io nei casi in analisi ero in possesso di dati già presentati con la medesima scala; avendo la necessità di individuare gli autovettori e applicare il criterio di Keiser nel trainset ho dovuto procedere ugualmente a standardizzare.
4. Calcolo della **PCA** con la seguente formula:

```
prcomp(df_numeric, scale = FALSE)
```

R offre due metodi per calcolare la PCA:

- `prcomp(x, scale = FALSE)`: dove x rappresenta una matrice numerica o data frame e `scale` un valore logico che indica se le variabili devono essere ridimensionate/standardizzate;

- *princomp*(*x*, *cor* = *FALSE*, *scores* = *TRUE*): dove *x* rappresenta una matrice numerica o data frame, *cor* valore logico che se a true ridimensiona e centra i dati prima di procedere all'analisi e *scores* valore logico che se a true calcola le coordinate su ciascun componente principale.

Metodologia che ho utilizzato per passi:

Ho deciso di far uso della funzione *prcomp* perchè usa la decomposizione del valore singolare (SGV) che offre una precisione leggermente migliore rispetto all'uso del metodo *princomp*. Il metodo *prcomp* include nei propri elementi di output:

- (a) *sdev*: deviazione standard delle componenti principali;
- (b) *rotation*: la matrice dei carichi delle variabili ovvero le colonne degli autovettori;
- (c) *center*: la media variabile, indica se le variabili devono essere spostate per essere centrate sullo zero;
- (d) *scale*: deviazione standard delle variabili;
- (e) *x*: coordinate degli individui sulle componenti principali.

5. Calcolo degli **autovalori della matrice di covarianza**. Mostra la percentuale di varianze di competenza di ciascun componente principale con

```
get_eig(res.pca)
```

6. **Riepilogo** mediante il metodo *summary* dei risultati ottenuti dal calcolo della PCA. Effettua la standardizzazione dei risultati ottenuti e ne calcola gli autovalori di covarianza con individuazione di deviazione standard, proporzione della varianza e proporzione cumulata.

```
summary(res.pca)
```

7. **Calcolo degli autovettori** con

```
loadings(res.pca)
```

8. **Individuazione della matrice correlazione** dei dati con

```
cor(df_numeric)
```

9. Creazione di un data frame contenete per ogni variabile principale quali sono le componenti ordinate in modo decrescente strettamente correlate.

3.3 Risultati ottenuti

I dati risultanti dall'elaborazione dei dati inerenti il trainset delle domande nel database gli ho riportati, per motivi di spazio e comprensione, in modo parziale. Ogni metodo utilizzato viene di seguito descritto nel dettaglio.

3.3.1 Calcolo della Principal Component Analysis

```
> prcomp(df_numeric, scale = FALSE)
Standard deviations (1, ..., p=6):
[1] 1.5905164 0.9270882 0.6549647 0.2877259 0.2567623 0.2267723

Rotation (n x k) = (6 x 6):
      PC1      PC2      PC3      PC4      PC5      PC6
V1 -0.3136368 -0.06351095 0.6506652 -0.0008078537 0.660023581 0.196472324
V2 0.3888043 0.53581153 0.2479470 0.7072146111 -0.008846399 0.005359089
V3 -0.5186977 0.45305827 -0.1486278 0.0011520653 0.150101946 -0.693591199
V4 -0.3009414 -0.07357741 0.6112341 -0.0012775235 -0.719969817 -0.109796655
V5 0.3876316 0.53818077 0.2452635 -0.7069936980 -0.008478550 0.006089356
V6 -0.4905017 0.45668194 -0.2435273 0.0019475491 -0.152746837 0.684257823
```

Figura 17: Visualizzazione del calcolo della PCA tramite prcomp per il trainset di prova.

```
> prcomp(df_numeric, scale = FALSE)
Standard deviations (1, ..., p=89):
[1] 0.50804395 0.55848746 0.54971833 0.53126009 0.52009133 0.51577310 0.50144770 0.49349655 0.48596238 0.47999062 0.47018338 0.46401672 0.45867619 0.44999542 0.44252412 0.43580806 0.43223385 0.42680299
[19] 0.42079248 0.41533049 0.41068652 0.40526268 0.40119893 0.39576359 0.39038038 0.38494451 0.38161779 0.37446709 0.37095118 0.36811364 0.36109314 0.35803208 0.35352538 0.34981222 0.34587283
[37] 0.34221823 0.33861509 0.33532453 0.33275471 0.32925297 0.32449961 0.32272528 0.31788589 0.31716223 0.31403458 0.30962208 0.30523877 0.30243220 0.29816970 0.29458133 0.29251356 0.29083447 0.28424146
[55] 0.28177829 0.28074860 0.27888498 0.27585038 0.27422473 0.27139694 0.27019256 0.26679640 0.26427259 0.26314999 0.25905281 0.25708924 0.25674716 0.25387156 0.25253431 0.24954843 0.24591523 0.24355894
[73] 0.24107129 0.23709184 0.23498756 0.23280141 0.22959880 0.22846454 0.22343120 0.19833321 0.18012511 0.17449777 0.15236893 0.14016744 0.13282731 0.10140180 0.08865511 0.02703589 0.02334543

Rotation (n x k) = (89 x 89):
      PC1      PC2      PC3      PC4      PC5      PC6      PC7      PC8      PC9      PC10     PC11     PC12     PC13     PC14
V1 2.636387e-03 0.0053609165 -4.507406e-03 0.0047285077 -6.267813e-04 0.0012179541 -0.0006078271 0.0015559397 -1.333849e-03 0.0001061671 -0.0037520794 0.0133571060 -0.0124606511 0.016606655
V2 1.180602e-01 -0.0983273120 -2.06575e-01 -0.1595809648 1.666802e-01 -0.5078924212 -0.4156870156 -0.0055475540 2.878860e-01 -0.2664054093 -0.2141239846 -0.1154210418 -0.2867785700 -0.004759859
V3 2.418395e-03 -0.0075282117 1.23714e-02 -0.0130787491 3.664082e-03 -0.0173104077 0.0139035261 -0.0076250979 -1.464138e-02 -0.0268438815 -0.0315897238 0.0122222675 0.0332085902 -0.007904186
V4 -6.730258e-03 -0.0157451591 -2.170800e-02 -0.0117814649 4.12775e-02 0.0022788941 -0.0080299230 -0.0340036671 7.105564e-03 0.0206008584 -0.0081277662 0.0265778032 0.0022726268 -0.007387800
V5 1.514325e-02 -0.0057397594 -6.121892e-03 0.0045128218 5.433210e-03 0.0006548622 -0.0105138277 -0.0003915063 5.902137e-03 0.0027513825 0.0090012403 0.0390923132 0.0001094258 -0.007355540
V6 -3.397820e-02 -0.0335866990 1.107013e-02 -0.0932617778 2.604523e-05 0.0141769088 0.0067276127 -0.0041570833 -2.533817e-02 -0.0384170573 0.0118988680 -0.0188097858 0.0477093376 0.022818524
V7 1.314959e-02 -0.0070974227 -3.653409e-02 0.0332109315 4.940004e-02 -0.0255328018 -0.0201294406 -0.040800858 7.001952e-02 -0.0055010060 -0.0186226671 -0.0534379835 -0.005495458 0.009432314
V8 6.752564e-02 0.1271215300 -9.607290e-02 0.2670813592 2.783064e-02 -0.0864297064 0.0028403117 0.0012703572 -4.578910e-02 -0.0252448861 0.0348617417 -0.0236246302 -0.0258508008 -0.170862744
V9 4.916774e-02 0.0313629889 -6.180563e-02 0.1604337086 1.244642e-01 0.0159534435 0.0340172949 -0.0240708166 -7.438052e-02 -0.1016247255 -0.0391609350 -0.1892908371 0.1526616739 0.044115389
V10 1.203606e-02 -0.0068310303 -1.251339e-02 -0.0234524814 -2.780708e-02 -0.0023647650 -0.0193398045 -0.0257208548 3.119853e-02 0.0300631227 -0.0222724247 0.0529256183 -0.0389295281 -0.029166357
V11 -4.576335e-02 -0.0376339650 -7.111440e-02 -0.1541784131 -6.025191e-02 0.0093648752 -0.0358636247 -0.1687252230 -1.255892e-01 0.3334747609 -0.1520333807 -0.0680811778 -0.0094640942 -0.326270930
V12 1.707129e-02 0.1619149072 -1.751043e-01 -0.2636588899 -1.072001e-01 0.0471839798 0.0078876045 -0.0047098767 -1.602491e-01 -0.0972289384 0.1299995080 0.0078102762 0.0760172522 -0.007572510
V13 2.411796e-04 -0.0088153060 -1.048016e-02 -0.0395492127 -4.480856e-02 0.0199669461 0.0252357671 0.0631595225 8.353414e-03 -0.0116540106 0.0464408945 0.0549463204 0.0400919256 0.049566387
```

Figura 18: Visualizzazione del calcolo della PCA tramite prcomp per il trainset delle domande nel database.

Osservazioni Osservando esclusivamente la deviazione standard ottenuta dal trainset di prova, appare come i PC da prendere in considerazione per l'analisi sono i primi tre; inoltre nella rotazione appare chiaro che le valutazioni ottenute da V1 sono in relazione con V4, V3 con V6 e V2 con V5. Tuttavia queste sono solo mere osservazioni senza ancora alcuna prova matematica completa a supporto.

Per quanto concerne il trainset delle domande nel database è molto difficile fare qualunque tipo di assunzione sulla natura dei dati causa la loro numerosità.

3.3.2 Calcolo degli autovettori ed individuazione di Summary

```
> summary(res.pca)
Importance of components:

               Comp.1   Comp.2   Comp.3   Comp.4   Comp.5   Comp.6
Standard deviation   1.9297771 1.0581456 0.9099156 0.38288957 0.34464930 0.244856110
Proportion of Variance 0.6209837 0.1867054 0.1380601 0.02444629 0.01980709 0.009997418
Cumulative Proportion 0.6209837 0.8076891 0.9457492 0.97019549 0.99000258 1.000000000
> # fornisce gli autovalori della matrice di covarianza/varianza delle dimensioni principali
> get_eig(res.pca)

eigenvalue variance.percent cumulative.variance.percent
Dim.1 3.72403947      62.0983737                62.09837
Dim.2 1.11967213      18.6705374                80.76891
Dim.3 0.82794633      13.8060084                94.57492
Dim.4 0.14660442       2.4446293                97.01955
Dim.5 0.11878314       1.9807094                99.00026
Dim.6 0.05995451       0.9997418                100.00000
```

Figura 19: Visualizzazione del calcolo del metodo summary ed individuazione degli autovalori per il trainset di prova.

```
> summary(res.pca)
Importance of components:

               Comp.1   Comp.2   Comp.3   Comp.4   Comp.5   Comp.6   Comp.7   Comp.8   Comp.9   Comp.10   Comp.11   Comp.12   Comp.13   Comp.14   Comp.15   Comp.16
Standard deviation   1.43136144 1.30933960 1.27248819 1.25540778 1.24609579 1.22900589 1.22206570 1.21814257 1.20907848 1.19278209 1.17999983 1.17045380 1.16024099 1.15735299 1.15660352 1.14908663
Proportion of Variance 0.02303868 0.01927807 0.01820818 0.01772265 0.01746071 0.01698505 0.01679377 0.01668611 0.01643872 0.01599857 0.0156336 0.01548521 0.01534701 0.01506696 0.01504486 0.01484788
Cumulative Proportion 0.02303868 0.04231675 0.06052493 0.07824758 0.09570828 0.11269334 0.12948710 0.14617321 0.16261193 0.17861059 0.1942441 0.20964931 0.22499632 0.24006328 0.25510814 0.26995602

               Comp.17   Comp.18   Comp.19   Comp.20   Comp.21   Comp.22   Comp.23   Comp.24   Comp.25   Comp.26   Comp.27   Comp.28   Comp.29   Comp.30   Comp.31   Comp.32
Standard deviation   1.13838196 1.12889068 1.12429911 1.11484336 1.10634870 1.10687053 1.09819751 1.09140282 1.0838406 1.07913609 1.07706163 1.07078478 1.06286985 1.05672587 1.05326482 1.05119497
Proportion of Variance 0.01457253 0.01433055 0.01421421 0.01397612 0.01376395 0.01375703 0.01356188 0.01339458 0.0132096 0.01309518 0.01304488 0.01289328 0.01270338 0.01255693 0.01247482 0.01242583
Cumulative Proportion 0.28452855 0.29885910 0.31307331 0.32704043 0.34081338 0.35457041 0.36813229 0.38152607 0.3947355 0.40783165 0.42087053 0.43376981 0.44647318 0.45903012 0.47150493 0.48393076

               Comp.33   Comp.34   Comp.35   Comp.36   Comp.37   Comp.38   Comp.39   Comp.40   Comp.41   Comp.42   Comp.43   Comp.44   Comp.45   Comp.46   Comp.47   Comp.48
Standard deviation   1.04333725 1.04116605 1.02871449 1.02479858 1.02307950 1.01728386 1.01558854 1.00733966 1.00662110 0.99744297 0.99650091 0.99578402 0.98376459 0.98205540 0.97349201 0.97125204
Proportion of Variance 0.01224076 0.01189987 0.01190005 0.01180962 0.01177093 0.01163706 0.01159812 0.01141066 0.01139439 0.01118755 0.01116643 0.01114858 0.01088282 0.01084503 0.01065672 0.01060774
Cumulative Proportion 0.49617152 0.50836139 0.52065144 0.53207106 0.54384509 0.55547815 0.56707628 0.5788604 0.58988133 0.60106088 0.61223531 0.62338389 0.63426671 0.64511174 0.65576847 0.66637621

               Comp.49   Comp.50   Comp.51   Comp.52   Comp.53   Comp.54   Comp.55   Comp.56   Comp.57   Comp.58   Comp.59   Comp.60   Comp.61   Comp.62   Comp.63
Standard deviation   0.96534887 0.95952978 0.95755890 0.95339737 0.94198469 0.93825159 0.93238471 0.93115086 0.92138027 0.91825097 0.91639097 0.90715173 0.90122820 0.89453214 0.89306972
Proportion of Variance 0.01047919 0.01031323 0.01031074 0.01022132 0.00997037 0.00999914 0.00977405 0.00974054 0.00974059 0.00954468 0.00948263 0.00944123 0.00923373 0.00913317 0.00899831 0.008968704
Cumulative Proportion 0.67685539 0.68728862 0.69751937 0.70774068 0.71771706 0.72761620 0.73739025 0.74714015 0.75668480 0.76616639 0.77560963 0.78486340 0.79399671 0.80299481 0.811963523

               Comp.64   Comp.65   Comp.66   Comp.67   Comp.68   Comp.69   Comp.70   Comp.71   Comp.72   Comp.73   Comp.74   Comp.75   Comp.76   Comp.77   Comp.78
Standard deviation   0.88788602 0.88008493 0.87785261 0.87186920 0.86541478 0.85970807 0.85674123 0.84783617 0.84675815 0.83708897 0.82968803 0.82365298 0.81735535 0.81241108 0.80877804
Proportion of Variance 0.00888612 0.00870979 0.00864984 0.00847444 0.00842181 0.00831147 0.00823383 0.00803191 0.00806249 0.00787564 0.00774889 0.00762846 0.00751209 0.00742182 0.00735547
Cumulative Proportion 0.82862816 0.82953615 0.83818649 0.84673442 0.85515629 0.86346740 0.87172132 0.87980451 0.88786764 0.89574678 0.90348756 0.91111622 0.91862831 0.92605012 0.93340572

               Comp.79   Comp.80   Comp.81   Comp.82   Comp.83   Comp.84   Comp.85   Comp.86   Comp.87   Comp.88   Comp.89
Standard deviation   0.79836916 0.79614608 0.78236719 0.77255190 0.75573703 0.75517304 0.74555534 0.73810154 0.71756860 0.69310962 0.64490760
Proportion of Variance 0.00716748 0.00712729 0.00688383 0.00671141 0.00649061 0.00641263 0.00621702 0.00596132 0.00579069 0.00540210 0.00243041
Cumulative Proportion 0.94057305 0.94770681 0.95458403 0.96129542 0.96778693 0.97419856 0.98041618 0.98637734 0.99216713 0.99756919 1.00000000
```

Figura 20: Visualizzazione del calcolo del metodo summary per il trainset delle domande nel database.

```
> get_eig(res.pca)
```

	eigenvalue	variance.percent	cumulative.variance.percent
Dim.1	2.0487956	2.3038680	2.303868
Dim.2	1.7143702	1.9278071	4.231675
Dim.3	1.6192262	1.8208178	6.052493
Dim.4	1.5760487	1.7722647	7.824758
Dim.5	1.5527547	1.7460707	9.570828
Dim.6	1.5104555	1.6985053	11.269334
Dim.7	1.4934446	1.6793765	12.948710
Dim.8	1.4838713	1.6686114	14.617321
Dim.9	1.4618708	1.6438718	16.261193
Dim.10	1.4227291	1.5998571	17.861050
Dim.11	1.3902728	1.5633600	19.424410
Dim.12	1.3699621	1.5405206	20.964931
Dim.13	1.3647870	1.5347012	22.499632
Dim.14	1.3398826	1.5066963	24.006328
Dim.15	1.3379168	1.5044857	25.510814
Dim.16	1.3204001	1.4847882	26.995602
Dim.17	1.2959135	1.4572531	28.452855
Dim.18	1.2743942	1.4330546	29.885910
Dim.19	1.2640485	1.4214209	31.307331
Dim.20	1.2428757	1.3976122	32.704943
Dim.21	1.2240074	1.3763948	34.081338
Dim.22	1.2233920	1.3757027	35.457041
Dim.23	1.2060378	1.3561879	36.813229
Dim.24	1.1911601	1.3394580	38.152687
Dim.25	1.1747105	1.3209604	39.473647
Dim.26	1.1645347	1.3095178	40.783165
Dim.27	1.1600618	1.3044880	42.087653
Dim.28	1.1465801	1.2893278	43.376981
Dim.29	1.1296923	1.2703376	44.647318
Dim.30	1.1166696	1.2556935	45.903012
Dim.31	1.1093668	1.2474815	47.150493
Dim.32	1.1050109	1.2425833	48.393076
Dim.33	1.0885526	1.2240760	49.617152

Figura 21: Individuazione degli autovalori per il trainset delle domande nel database.

Osservazioni È essenziale per individuare il numero di componenti (PC) necessarie per effettuare un'analisi corretta basarsi sul calcolo della *variance.percent* o *Proportion of Variance*. A tale scopo è prendere le componenti principali che catturano la maggior parte di variabilità dei dati. Nel caso del trainset di prova basta le variabili PC1, PC2 e PC3 sono sufficienti a catturare il 93% della variabilità presente. Quanto appena descritto si può riscontrare anche graficamente, come presentato di seguito.

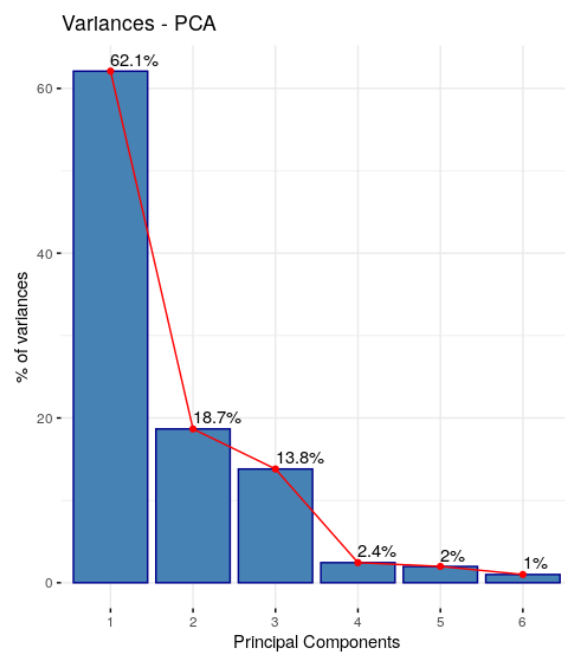


Figura 22: Plot della rappresentazione grafica della varianza sui PC del trainset di prova.

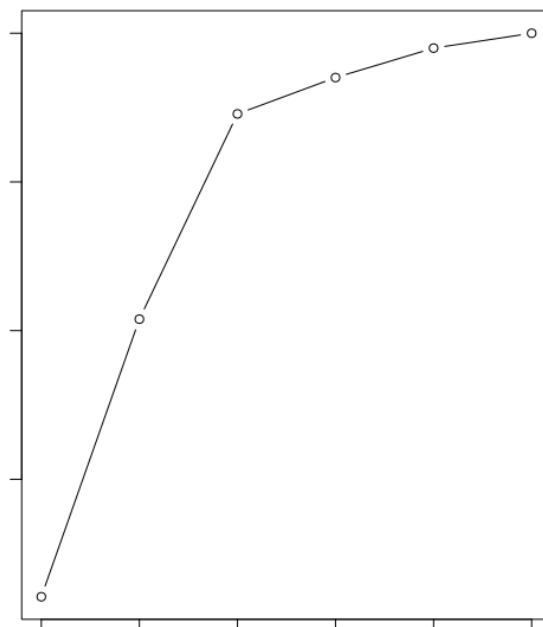


Figura 23: Plot della rappresentazione grafica della varianza sui PC del trainset di prova.

Le ultime PC4, PC5 e PC6 hanno una variabilità molto bassa, trascurabile.

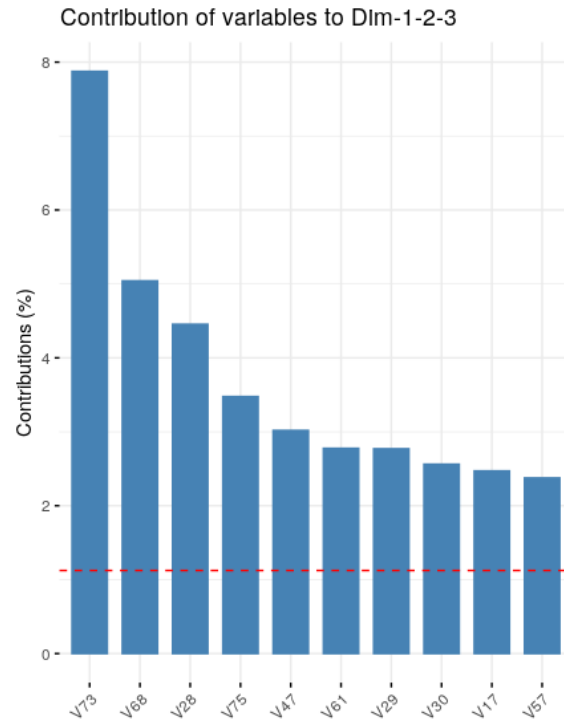


Figura 24: Rappresentazione grafica di come la varianza si distribuisce sulle PC individuate dal modello sul trainset di prova.

Tuttavia per quanto riguarda le domande nel database ogni conclusione "a occhio " risulta impossibile da effettuare sempre a causa della numerosità dei dati di trainset. L'utilizzo di un analisi dei risultati per mezzo di plot è l'unica via percorribile.

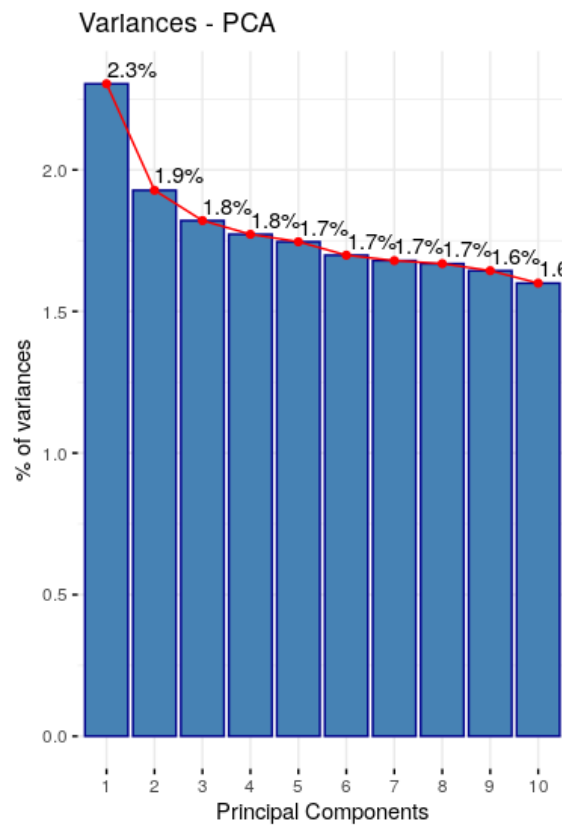


Figura 25: Plot della rappresentazione grafica della varianza dei primi dieci PC del trainset delle domande nel database.

Il plot mostra esclusivamente le prime dieci componenti; tutte si presentano con una varianza molto basse. A causa di ciò per poter affermare quante PC sono indispensabili per una valutazione oggettiva dei dati è indispensabile avere una visione totalitaria di tutte variabili coinvolte nel modello.

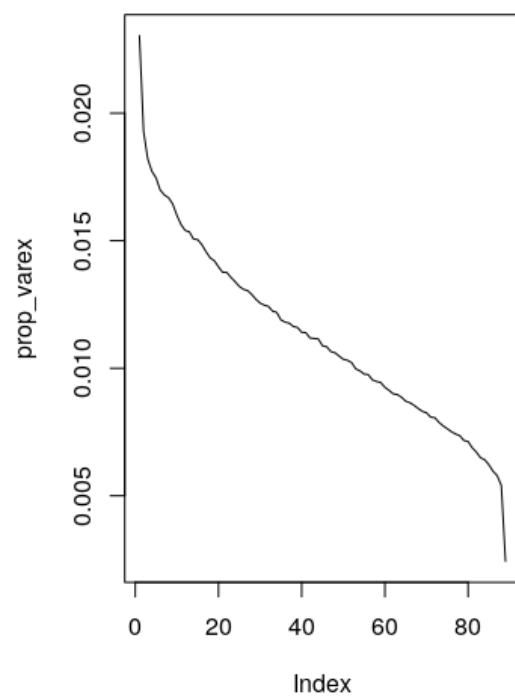


Figura 26: Plot della rappresentazione grafica della varianza di tutti i PC del trainset delle domande nel database.

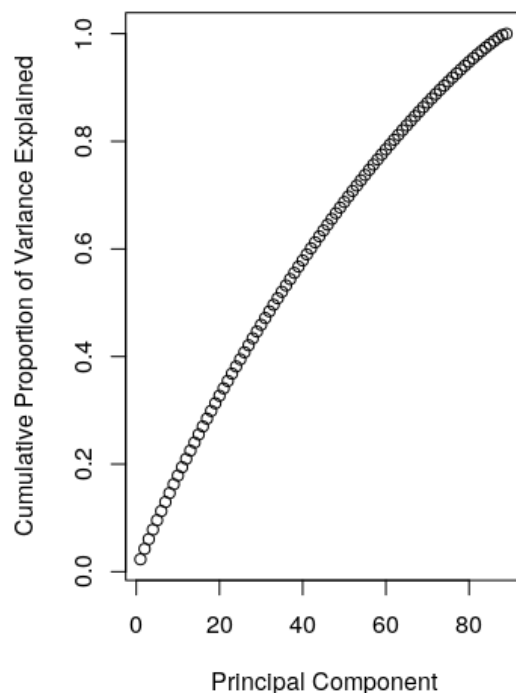


Figura 27: Plot della rappresentazione grafica della varianza di tutti i PC del trainset delle domande nel database.

Non risulta sufficiente l'andamento dei plot per riuscire a definire il numero adeguato di componenti principali da utilizzare. In questi casi, è buona norma, fare riferimento a tre criteri:

- *Quota della varianza totale*: si deve considerare un numero di CP tale che tenga conto di una percentuale sufficientemente elevata di varianza totale proporzionale al numero di variabili originarie (ovvero più è alto il numero di componenti del modello e più è accettata una percentuale minore di varianza spiegata).
- *Screen-graph*: fa uso dei plot degli autovalori in funzione al numero di CP. Gli autovalori sono decrescenti, per cui il grafico ha una buona possibilità, di assumere la forma di una spezzata con pendenza negativa.
- *Eigenvalue one o Regola di Kaiser*: afferma di considerare tutte ed esclusivamente le CP con autovalore maggiore di 1.

Per soddisfare il *primo criterio* è sufficiente fare riferimento a *variance.percent* in `get_eigen` o *Proportion of Variance* in `summary`. Da questa asserzione ne consegue un quesito: quale è il numero di varianza accettabile avendo un numero di variabili molto elevato?. Per effettuare una delle vie utilizzate è procedere al soddisfacimento del *terzo criterio*. Gli autovalori di tutte le componenti coinvolte si possono vedere nella *Standard deviation* risultante dalla `summary`. Le PC del modello, con autovalore superiore a 1, sono le PC contenute nell'intervallo 1-41. Il *secondo criterio*, invece, in questo caso specifico non è stato in grado di dirmi molto, in quanto la diminuzione degli autovalori è graduale, senza salti evidenti. In conclusione, ho considerato come una percentuale di copertura adeguata quella fornita dalla che 41 prime componenti principali.

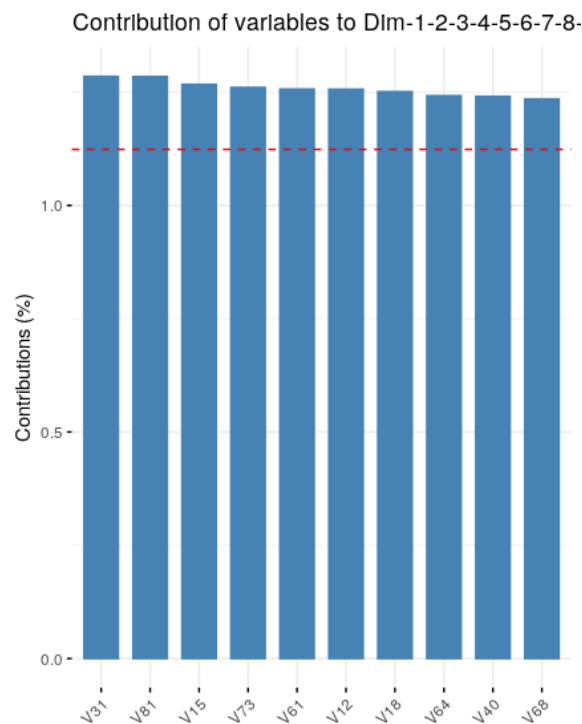


Figura 28: Rappresentazione grafica di come la varianza si distribuisce sulle PC individuate dal modello sul trainset delle domande nel database.

3.3.3 Calcolo degli autovettori

```
> loadings(res.pca)

Loadings:
      Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
V1  0.420      0.569  0.702
V2 -0.394  0.515  0.282      -0.707
V3  0.418  0.477 -0.278      -0.721
V4  0.421      0.565 -0.709
V5 -0.393  0.518  0.279      0.707
V6  0.402  0.489 -0.350      0.689

      Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
SS loadings  1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var 0.167  0.167  0.167  0.167  0.167  0.167
Cumulative Var 0.167  0.333  0.500  0.667  0.833  1.000
```

Figura 29: Autovettori del trainset di test.

Osservazioni L'analisi dei loadings sulle componenti principali permette di determinare il contributo delle variabili originarie al modello PC.

Per quanto riguarda l'analisi dei dati del trainset la prima variabile ha valori 0.420, 0.569 e 0.702, rispettivamente nelle componenti 1, 3 e 4 e sembra legarsi ai risultati nella variabile 4 (che riempie le medesime componenti oscillando di poco nei valori presentati), il medesimo match coinvolge le variabili 2 con 5 e 3 con 6.

Tali assunzioni sono molto più complesse da effettuare per il trainset dei dati delle domande nel databas e per fare ciò ho provveduto a calcolare la matrice correlazione.

3.3.4 Calcolo della matrice di correlazione

```
> cor.pca
      V1      V2      V3      V4      V5      V6
V1  1.0000000 -0.4725415  0.5354458  0.8534648 -0.4724833  0.4749618
V2 -0.4725415  1.0000000 -0.4025775 -0.4839185  0.8811501 -0.3892455
V3  0.5354458 -0.4025775  1.0000000  0.5237420 -0.3996431  0.9371832
V4  0.8534648 -0.4839185  0.5237420  1.0000000 -0.4838589  0.4736120
V5 -0.4724833  0.8811501 -0.3996431 -0.4838589  1.0000000 -0.3862259
V6  0.4749618 -0.3892455  0.9371832  0.4736120 -0.3862259  1.0000000
```

Figura 30: Autovettori del trainset di test.

> cor_pca	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
V1	1.000000000	0.020272106	-0.006563494	-0.009038249	0.032402323	-1.22265e-02	1.440978e-02	0.007078308	-0.0649519219	0.0096352727	0.0025737977	0.006867180	-1.203807e-02	0.0062179747
V2	0.02027211	1.000000000	0.0109484299	0.0277994172	0.0022019004	-1.355124e-02	7.014736e-02	0.0149970844	-0.0153076421	0.0310324650	-0.0327089265	-0.010560986	-5.563379e-02	-0.0187787934
V3	-0.004563494	0.0109484299	1.000000000	0.0099002028	0.0035048287	-5.024745e-03	-1.744317e-02	0.0096161144	0.0290833578	-0.018684332	-0.049945853	0.011805678	4.296537e-03	-0.0123475985
V4	-0.009383294	0.0277994172	0.0099002028	1.000000000	0.0070999842	-2.440739e-04	5.624869e-02	0.0022587458	0.0363071171	0.0152464032	0.0338330601	0.0355881260	-1.261759e-02	0.0236108359
V5	0.032402326	0.0022019004	0.0035048287	0.0070999842	1.000000000	9.404832e-03	1.852404e-02	-0.0055583816	0.0257313911	-0.0075689377	-0.0020809503	-0.006241190	1.000551e-02	-0.0048544805
V6	-0.01222653	-0.013551263	-0.0058247451	-0.0002448739	0.0096045826	1.000000e-00	-9.60535e-03	-0.0244332634	0.0086920338	-0.0080880666	-0.0655248602	0.037934184	0.639492e-03	-0.0690937292
V7	0.014499783	0.070173627	-0.0174431726	0.0562486863	0.0185240363	-9.605935e-03	1.000000e+00	0.0581811046	0.0185490010	0.0156306093	0.0231381746	-0.0031088528	-5.174636e-02	0.0362460823
V8	0.00707831	0.0149970844	0.0096161144	-0.0022587458	-0.0055583816	-2.443236e-02	5.618110e-02	1.0000000000	0.0765809911	0.0078533444	0.0356712371	-0.0263441964	-2.351978e-02	-0.0004265940
V9	-0.064951922	-0.0153076421	0.0290833538	0.0365071171	0.0257313911	0.0492034e-03	1.854900e-02	0.0766580911	1.0000000000	-0.0349393777	-0.0142172292	-0.0101187252	-2.265859e-02	-0.0064913600
V10	0.009635273	-0.0327089265	-0.0186844323	0.0152464032	-0.0075689377	-8.088067e-03	1.563069e-02	0.0078533144	-0.0349393777	1.0000000000	0.0527671384	0.0174195214	-1.752807e-02	-0.0135164556
V11	0.002557398	-0.0327089285	-0.0499445853	0.0338330601	-0.0020809503	-6.552641e-02	2.113817e-02	0.0256721371	-0.0142172292	0.0527671384	1.0000000000	0.0086376189	-9.989418e-02	0.0476110206
V12	0.006867118	-0.0105609406	0.0118058675	0.0355881260	-0.006241190	3.779342e-02	-3.108653e-03	-0.0263441964	-0.0101187252	0.0174195214	0.0086376189	1.0000000000	-2.087564e-02	-0.0436317908
V13	-0.012347598	-0.055537800	0.0042965374	-0.0126175820	0.0100850666	0.639492e-03	-5.174636e-02	-0.0235197807	-0.0226585934	-0.0175289660	-0.0998940992	-0.0287565354	1.000000e+00	-0.0118652383
V14	0.006179775	-0.0187787934	-0.0123475985	0.0236108359	-0.0048544805	-9.009373e-02	3.624695e-02	-0.0084265948	-0.0084913600	-0.0315645566	0.0476110206	-0.0436317908	-1.186524e-02	1.0000000000
V15	-0.014758925	-0.0100939765	0.0264112927	-0.0093946234	-0.0238124172	-3.009299e-02	1.114677e-02	0.0207706267	-0.0392091147	-0.022212800	-0.0017077176	0.0361403393	-6.254951e-02	-0.0093746618
V16	-0.026626358	-0.011545265	-0.0565730729	-0.0469258098	0.0042767939	1.902076e-03	1.274478e-03	0.0114727247	-0.0170850337	-0.0376931682	-0.0025752982	-0.0019073041	-4.270461e-02	-0.0735133872
V17	-0.022553887	0.019581396	0.0232952095	0.0151158566	0.0135916823	-0.883897e-02	-1.720176e-02	0.0732774112	0.0398040762	-0.0153210876	0.0053478215	-0.0043380998	-0.066858e-02	0.0243573872
V18	0.031959719	-0.015294780	-0.0282878561	0.0568743229	-3.990590e-03	4.480676e-02	-0.0312988401	0.0308294586	-0.0086538392	0.0232946240	-0.0251171795	7.200024e-03	0.0003392265	0.0003392265
V19	0.07722819	0.061647697	0.0205788512	-0.0260978417	0.0235080000	1.955237e-02	5.402097e-03	-0.0094704928	-0.0013309207	0.0156482252	-0.0138078604	0.0263253378	6.487972e-02	-0.0178439408
V20	-0.104791476	0.0496297021	-0.0035088286	-0.006636388	-0.144212569	-8.977787e-03	-5.282266e-02	0.0051956413	-0.0043515419	0.0070749883	0.0063787773	-0.0047326e-03	0.0018376861	0.0004376582
V21	0.001301225	0.0316287927	-0.0186170659	0.0240760152	-0.0010221703	-1.603207e-02	6.974480e-02	0.0154436061	0.0165919004	-0.0139277357	0.0504173164	0.0490217371	-5.487736e-03	0.0508444458
V22	0.045962996	-0.0423900226	0.0268650232	-0.0098347467	0.0006265081	-4.037566e-02	7.071899e-03	0.0407826922	-0.0136334034	-0.0072720183	-0.0258236358	-0.0311195464	-3.022008e-02	-0.0307062098
V23	0.009565787	-0.0113444995	0.0444239593	-0.0161160728	-0.0078557749	5.642355e-02	-4.119466e-03	0.0143374178	-0.0122236390	-0.0277088011	-0.0205181241	-0.0428572731	-3.804852e-02	-0.0450139111
V24	0.020861554	-0.0078510543	0.0115072959	0.0104098602	-5.18751e-02	1.604833e-02	0.064833e-02	0.0522678520	0.0180121917	0.0098716182	-0.0209595968	-0.0309977118	-1.442688e-03	0.0308945664
V25	0.012735958	-0.035132590	-0.0423664687	-0.0102464149	0.0061111783	-4.992188e-02	2.756455e-02	0.0375284283	-0.0062977412	-0.0200749214	-0.0200813582	0.0080121762	-1.886551e-02	-0.0064415288
V26	-0.00076081	0.0437080951	-0.0204118478	-0.0192457096	0.0050902468	-2.641208e-02	-3.814904e-02	0.0091286432	-0.0047883208	-0.0022580353	0.0172217957	-0.0024253206	-3.725737e-02	-0.0100715181
V27	0.001595846	0.0052576995	-0.0289585675	-0.0389506607	-0.0015383680	-3.752493e-02	-4.667907e-02	0.0657728135	-0.0641613837	0.0119144912	0.0206141402	-0.0150782107	-0.0158762102	-0.0192883809
V28	0.006859805	0.0056454438	0.0016281298	0.0032246160	0.0042716139	4.362135e-03	-1.617656e-02	-0.0025244628	-0.0231730686	-0.0034376016	-0.0009124095	-0.00309993260	4.580558e-03	-0.0022847745
V29	-0.060480869	0.004008793	-0.005681786	-0.0036022137	0.024200271	9.377774e-03	-1.282100e-02	-0.0054271170	-0.0045454114	-0.0073001925	-0.0051650977	-0.0064620406	0.047331e-02	-0.0047330477
V30	0.08117332	0.024603493	0.0007919649	0.0100453533	-0.0582349537	1.558896e-02	-4.936794e-02	-0.0078642298	-0.075163816	-0.0107088482	-0.0096554881	-0.0096554881	1.426393e-02	-0.0068683339
V31	0.015643566	-0.018521426	-0.0145959643	-0.0218711370	-0.0112063393	-9.158019e-02	1.523266e-02	-0.0111650752	0.0325952484	-0.0395465069	-0.0292287789	-0.0236433864	-5.295272e-02	0.0140466496
V32	0.024608001	-0.0038272079	-0.0189354505	-0.0308029342	-0.0072042951	-3.782271e-02	2.223626e-02	0.0735069278	0.0488627662	-0.0496173499	0.0020800101	0.0206264840	-4.796123e-03	0.01355462879
V33	0.004874525	-0.014792427	0.0330943818	-0.0661715468	-0.0033007214	-4.091023e-02	6.711960e-02	0.0508077391	-0.0647437506	-0.0295168027	0.0313914808	-0.0001254868	-2.493290e-02	0.0401322686
V34	0.016266451	-0.0446870482	-0.002401855	0.0027019968	-0.0130680211	-1.589349e-02	-1.116960e-02	-0.0904235780	-0.0174180527	-0.0393713876	0.0083573090	0.0045885559	-3.054788e-02	-0.0469246356
V35	-0.040844583	-0.050629348	-0.0097860238	0.0067496206	0.0282693454	1.843719e-02	-6.455919e-02	-0.044088329	-0.0469380450	-0.0450905899	-0.0265444157	-0.0325695266	-0.0135157e-02	-0.0173156735
V36	-0.010432892	-0.0765002870	-0.0034043120	-0.017285893	0.0082041941	-3.007224e-02	1.356067e-02	0.0077480409	-0.0267110404	0.0514032884	0.0553080447	0.0331457637	-1.799453e-02	-0.0192302794
V37	0.009568084	-0.0150980825	0.0635131458	-0.0176091642	-0.0075811384	-2.354418e-02	-4.576751e-02	0.0035853161	0.0220809750	-0.0492936022	0.0062671749	-0.0298704657	-2.218006e-02	-0.053563615
V38	-0.009846157	-0.0236583883	-0.0339360622	0.0281073376	0.0230927681	1.929486e-02	-0.1486109e-02	-0.0224116215	0.0585327987	0.0066000548	-0.0129748364	0.0131085226	0.00609094e-03	-0.0242852181
V39	0.012515222	-0.029519701	-0.0027640807	-0.0548243741	-0.0095400355	4.961018e-03	1.438521e-03	-0.0173965515	-0.0191110908	-0.0348691071	0.0028995906	-0.0039131474	-0.012021e-02	0.0153979974
V40	0.045395919	-0.027233276	-0.017460806	-0.0206630799	-0.0346208446	-2.305083e-03	-3.861630e-02	-0.0433786369	-0.0378220985	-0.0384645639	-0.0520978138	-0.022386315	0.027160e-02	-0.0273261579
V41	0.011096954	-0.0374910888	-0.0134970214	-0.0077110271	-0.0087175139	1.582526e-03	0.8791390e-03	-0.0360435448	-0.0271766879	0.0253216326	-0.0275827018	-0.006476185	2.287136e-02	-0.026616937

Figura 31: Autovettori del trainset delle domande nel database.

I CSV generati di correlazione sono i seguenti:

	A
1	V1" V2" V3" V4" V5" V6
	"1 4 3 6 5 2
	"2 2 5 6 3 1 4
	"3 3 6 1 4 5 2
2	"4 4 1 3 6 5 2
	"5 2 6 3 1 4
	"6 6 3 1 4 5 2

Figura 32: CSV generato a partire dalla matrice correlazione del trainset della rete.

34" 34 40 52 38 26 54 36 48 69 39 84 16 19 1 73 46 20 59 40 7 64 32 72 11 83 68 75 82 88 57 12 19 89 4 96 81 58 43 56 71 66 42 3 70 29 21 61 41 74 29 85 5 23 87 6 9 30 47 78 13 77 50 67 76 22 27 79 10 49 2 51 24 14 80 17 45 37 63 35 31 25 55 62 44 33 15 53 8 39" 30 68 57 78 81 13 56 49 63 36 5 72 66 17 6 40 24 50 25 61 23 10 4 40 83 29 39 69 30 88 27 85 18 25 22 70 63 80 38 49 58 3 47 20 76 51 21 87 42 77 82 79 14 54 44 31 65 71 59 18 62 41 11 89 52 53 12 80 55 37 45 75 61 4 10 9 80 2 32 24 28 20 67 7 45 73 38" 30 64 85 11 12 73 76 74 95 52 23 64 34 59 57 14 50 44 7 32 18 30 49 76 40 31 38 5 29 5 47 41 79 28 65 72 52 82 71 15 69 70 7 8 48 68 83 86 58 21 46 43 20 62 30 73 51 1 24 87 45 26 68 17 4 13 19 67 61 88 80 61 69 58 63 7 6 53 18 33 42 37 22 27 73" 37 53 3 38 24 75 55 17 48 59 81 13 18 49 43 46 67 50 1 55 75 76 20 33 11 38 47 68 83 18 45 29 60 66 47 29 5 72 69 71 21 20 74 16 25 81 2 42 36 15 4 79 18 82 85 51 38 40 6 78 12 64 88 77 27 26 63 57 25 52 27 74 56 50 10 32 34 20 14 21 86 36 41 44 87 38" 38 81 9 30 40 18 34 73 1 65 4 29 82 50 2 5 6 23 69 51 7 62 12 56 33 61 45 22 13 10 77 27 46 59 21 29 41 80 43 49 55 64 60 26 67 68 83 42 79 52 74 20 36 35 73 1 63 27 11 88 87 54 19 48 86 31 8 66 14 53 16 47 30 84 25 3 44 24 81 76 78 58 70 57 75 17 32 80 15 38" 39 36 48 62 70 74 26 61 34 10 31 63 61 62 14 19 17 44 1 73 64 30 78 40 49 68 76 15 65 43 76 23 1 57 87 46 9 13 12 28 53 15 29 6 34 58 18 71 77 56 48 11 48 58 23 20 2 70 57 10 60 37 55 47 54 40 67 42 64 51 37 4 53 48 89 40" 40 74 13 29 30 1 38 28 70 87 54 41 55 89 64 80 28 50 17 34 16 36 83 24 84 38 98 19 69 43 74 25 32 38 78 24 76 85 31 72 57 46 42 62 70 47 68 63 75 77

con la domanda 1). Tale ragionamento vale per tutte le altre 5 componenti, i cui risultati rimangono coerenti con le aspettative e con ciò che viene dichiarato nel Grafo della conoscenza, rappresentato in §2.1.

Pensavo di poter effettuare il medesimo ragionamento anche per analisi dei dati del trainset del database, tuttavia questo non è stato possibile, a causa delle seguenti motivazioni:

- In primo luogo mi sono subito accorta che se una variabile si dichiara in correlazione con una componente, questa poi non è detto che sia in correlazione con la variabile;
- Analizzando il testo delle domande, contenute nel database aziendale ho riscontrato come il modello prodotto crea relazioni strette fra domande appartenenti trivialmente a categorie diverse, (ad esempio domande che trattano relazioni con altre di serie numeriche) e meno con domande che parlano dello stesso tema.
- Effettuando dei test nella Rete neurale costruita sul trainset delle domande nel database, non riuscivo a trovare alcun collegamento con le correlazioni che vengono evidenziata dal calcolo della PCA;
- Il plot delle variabili mostra una vicinanza più elevata con variabili diverse rispetto a quelle indicate come quelle maggiormente correlate nella matrice di correlazione.

A questo punto ho iniziato a farmi delle domande. Escludendo un mio errore di codifica (dopo aver opportunamente effettuato un accurato controllo del codice da me prodotto) e trovandomi di fronte ad una situazione dove i risultati inerenti ai dati del trainset di prova soddisfano appieno le attese (infatti non solo le correlazioni matchano perfettamente con quanto viene indicato dal grafo della conoscenza; ma anche il plot che viene generato dalla PCA sulle variabili presenta una coerenza stretta con tali assunzioni) ho iniziato a pensare che centrasse la possibilità che ha un candidato di indovinare correttamente una risposta ad una domanda.

Ho rifatto perciò tutta l'analisi su un nuovo modello adoperando il trainset di prova generato da n input che tengono conto anche della possibilità di indovinare. Come viene illustrato dai plots seguenti i dati individuati dal trainset di prova puro vengono totalmente falsati quando si tiene conto di tale eventualità. Questo è il motivo per cui la **Principal Component Analysis** non è in grado di dare dei risultati attendibili sul trainset delle domande del database, dove un candidato può aver risposto correttamente ad un quesito anche solo perchè ha "sparato a caso".

3.4 Conclusione dell'analisi

In conclusione quando si ha un test con domande a triplice scelta multipla un candidato ha elevata possibilità di indovinare le domande che non sa e il modello creato dalla PCA con i dati delle risposte alle domande presenti nel database ne sono la prova. Questo perchè se la possibilità di indovinare fosse bassa i risultati del modello ne verrebbero appena sporcati, e invece invalidano completamente la possibilità di avere un risultato attendibile e coerente con la realtà che permetta di costruire un Reticolo della Conoscenza.

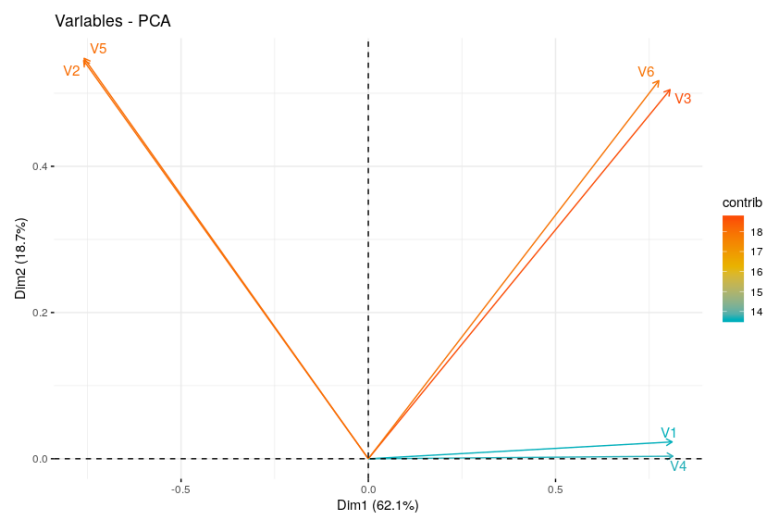


Figura 34: Rappresentazione per mezzo di plot di come le variabili si presentano nelle due componenti principali con il calcolo della PCA - utilizzo di trainset di prova puro sul grafo della conoscenza.

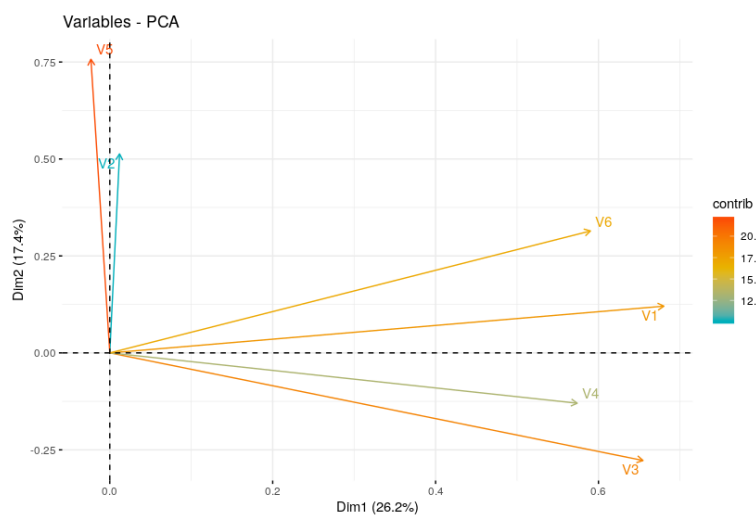


Figura 35: Rappresentazione per mezzo di plot di come le variabili si presentano nelle due componenti principali con il calcolo della PCA - utilizzo di trainset di prova spurio con la probabilità di indovinare.